第一章　STC11/10xx 系列单片机总体介绍

1.1　STC11/10xx系列 1T 单片机简介

STC11/10xx 系列单片机是宏晶科技设计生产的单时钟 / 机器周期(1T)的单片机，是高速 / 低功耗 / 超强抗干扰的新一代 8051 单片机，指令代码完全兼容传统 8051,但速度快 8-12 倍。内部集成高可靠复位电路，针对高速通信，智能控制，强干扰场合。

STC11/10xx系列单片机的定时器0/定时器1/串行口与传统8051兼容,增加了独立波特率发生器,省去了定时器2.传统 8051 的 111 条指令执行速度全面提速,最快的指令快 24 倍,最慢的指令快 3 倍.

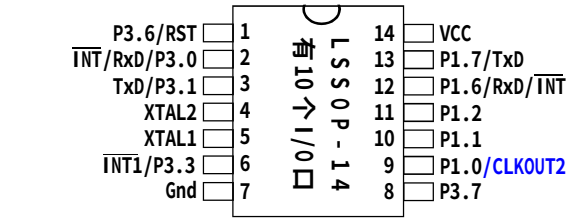
1. 增强型 8051 CPU，1T，单时钟 / 机器周期，指令代码完全兼容传统 8051
2. 工作电压：

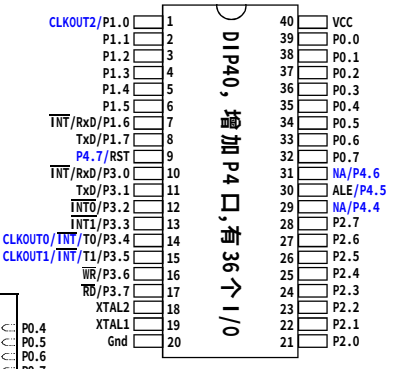
STC11Fxx 系列电压: 5.5V-4.1V/3.7V（5V 单片机）STC11Lxx 系列电压:3.6V-2.4V/2.1V（3V 单片机）
STC10Fxx 系列电压:5.5V-3.8V/3.4V（5V 单片机） STC10Lxx 系列电压:3.6V-2.4V/2.1V（3V 单片机）

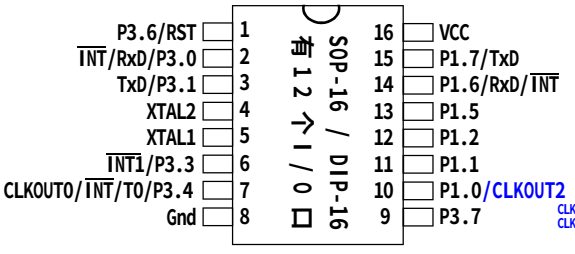
3. 工作频率范围：0 - 35MHz，相当于普通 8051 的 0～420MHz
4. STC11F/Lxx 系列单片机用户应用程序空间： 1/2/3/4/5/6/8/16/20/32/40/48/52/56/60/62K 字节
   STC10F/Lxx 系列单片机用户应用程序空间： 2K / 4K / 6K / 8K /10K / 12K / 14K 字节
5. STC11 系列单片机 ：RAM 为 1280 字节或 256 字节。STC10 系列单片机 ：RAM 为 512 字节或 256 字节
6. 通用 I/O 口（40/36 个），复位后为： 准双向口 / 弱上拉（普通 8051 传统 I/O 口）
   可设置成四种模式：准双向口 / 弱上拉，推挽 / 强上拉，仅为输入 / 高阻，开漏
   每个 I/O 口驱动能力均可达到 20mA，但整个芯片最大不要超过 100mA
7. ISP（在系统可编程）/ IAP（在应用可编程），无需专用编程器，无需专用仿真器
   可通过串口（RxD/P3.0，TxD/P3.1）直接下载用户程序，数秒即可完成一片
8. 有 EEPROM 功能
9. 看门狗
10. 内部集成 MAX810 专用复位电路(晶体频率在 24MHz 以下时，要选择高的复位门槛电压,如 4.1V 以下复位,晶体频率在 12MHz 以下时，可选择低的复位门槛电压，如 3.7V 以下复位，复位脚接 1K 电阻到地)
11. 内置一个对内部 Vcc 进行掉电检测的掉电检测电路，可设置为中断或复位
    5V 单片机掉电检测门槛电压为 4.1V/3.7V 附近，3.3V 单片机掉电检测门槛电压为 2.4V 附近
12. 时钟源：外部高精度晶体 / 时钟，内部 R/C 振荡器
    用户在下载用户程序时，可选择是使用内部 R/C 振荡器还是外部晶体 / 时钟
    常温下内部 R/C 振荡器频率为： 4MHz ～ 8MHz
    精度要求不高时，可选择使用内部时钟，但因为有制造误差和温漂，以实际测试为准
13. 2 个 16 位定时器(与传统 8051 兼容的定时器 / 计数器,16 位定时器 T0 和 T1),STC11xx/STC10xx 全系列都有 1 个独立波特率发生器（故不必用 T2 做为波特率发生器,详细使用方法请参考独立波特率发生器做串口通讯的相关使用说明及示例程序）
14. 3 个时钟输出口，可由 T0 的溢出在 P3.4/T0 输出时钟，可由 T1 的溢出在 P3.5/T1 输出时钟，
    独立波特率发生器可以在P1.0口输出时钟(部分型号无独立波特率发生器,详情请参阅单片机选型一览表)
15. 外部中断 I/O 口有 5 路,支持传统的下降沿中断或低电平触发中断
    Power Down(掉电)模式可由外部中断唤醒，INT0/P3.2，INT1/P3.3，INT/T0/P3.4，INT/T1/P3.5，INT/RxD/P3.0（或 INT/RxD/P1.6）
16. Power Down(掉电)模式可由内部掉电唤醒专用定时器唤醒(STC11xx 系列有此功能，STC10xx 无此功能)，也可由上面提到的外部中断口中断唤醒，由于 INT/RxD 支持下降沿中断，故也可支持远程通信唤醒
17. 一个独立的通用全双工异步串行口(UART)，做主机时可以当 2 个串口使用
    [RxD/P3.0，TxD/P3.1]可以切换到[RxD/P1.6，TxD/P1.7]，通过将串口在 P3 口和 P1 口之间来回切换，将 1 个串口作为 2 个主串口分时复用，可低成本实现 2 个串口，当然有其局限性
18. 工作温度范围： -40 - +85℃(工业级) / 0 - 75℃(商业级)
19. 封装:LSSOP14/SOP16/DIP16/DIP18/SOP20/DIP20/LSSOP20/PDIP-40/LQFP-44/PLCC-44(暂时尽量不要选PLCC44)
    SOP16/DIP16 有 12 个 I/O 口,SOP20/PDIP20/LSSOP20 有 16 个 I/O 口,LQFP44 有 40 个 I/O 口，PDIP40/QFN40(5mmx5mm)有 36 个 I/O 口

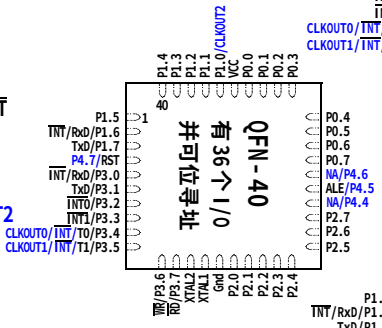
1.2 STC11/10xx系列单片机管脚图

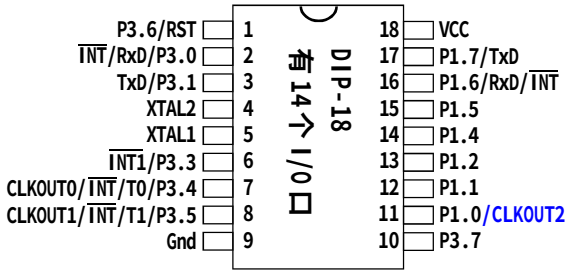
串行口做主机通信时，可控制串口通信在[RxD/P3.0,TxD/P3.1]和[RxD/P1.6,TxD/P1.7.]之间任意切换，实现 2 组串口。建议用户将自己的串行口设置在[RxD/P1.6,TxD/P1.7.]而将[RxD/P3.0,TxD/P3.1]口作为 ISP 下载的专用通信口,当然也可以当用户的普通 I/O 口用.如将复位脚 /RST 当 I/O 口使用，必须使用外部时钟

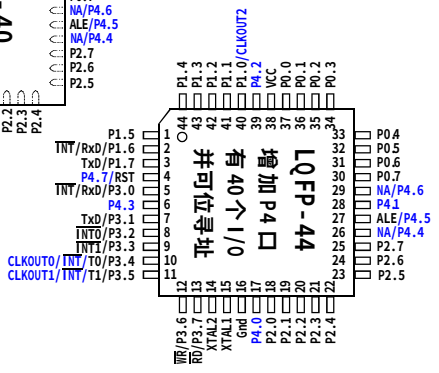


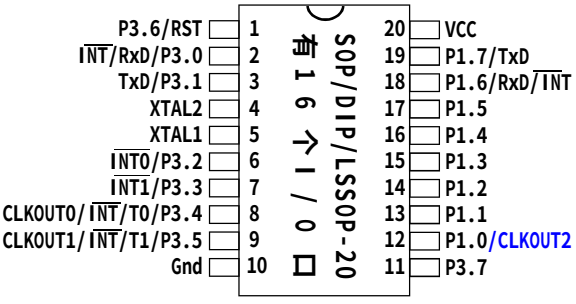


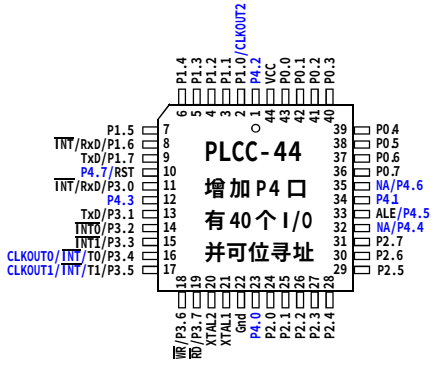












由 P4SW 寄存器设置(NA/P4.4，ALE/P4.5，NA/P4.6)三个端口的第二功能

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
P4SW	BBh	Port - 4 switch	-	NA_P4.6	ALE_P4.5	NA_P4.4	-	-	-	-	x000,xxxx

NA/P4.4: 0，复位后 P4SW.4 = 0,NA/P4.4 脚是弱上拉，无任何功能
1，通过设置 P4SW.4 = 1,将 NA/P4.4 脚设置成 I/O 口(P4.4)

ALE/P4.5: 0，复位后 P4SW.5 = 0,ALE/P4.5 脚是 ALE 信号,只有在用 MOVX 指令访问片外扩展器件时才有信号输出
1，通过设置 P4SW.5 = 1,将 ALE/P4.5 脚设置成 I/O 口(P4.5)

NA/P4.6: 0，复位后 P4SW.6 = 0,NA/P4.6 脚是弱上拉，无任何功能
1，通过设置 P4SW.6 = 1 将 NA/P4.6 脚设置成 I/O 口(P4.6)

在 ISP 烧录程序时设置 LQFP44/PDIP40/PLCC44 封装的单片机 RST/P4.7 管脚的第二功能，
RST/P4.7 在 ISP 烧录程序时选择是复位脚还是 P4.7 口，如设置成 P4.7 口，必须使用外部时钟。
在 ISP 烧录程序时设置 20Pin/18Pin/16Pin 封装的单片机 RST/P3.6 管脚的第二功能，
RST/P3.6 在 ISP 烧录程序时选择是复位脚还是 P3.6 口，如设置成 P3.6 口，必须使用外部时钟。

由 AUXR1 寄存器设置(串口 /UART)是在 P3 口还是在 P1 口

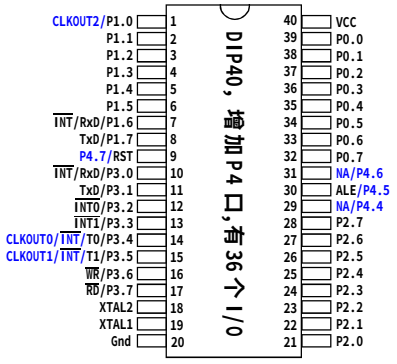
Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR1	A2h	Auxiliary register 1	UART_P1	-	-	-	GF2	-	-	DPS	0xxx,0x0

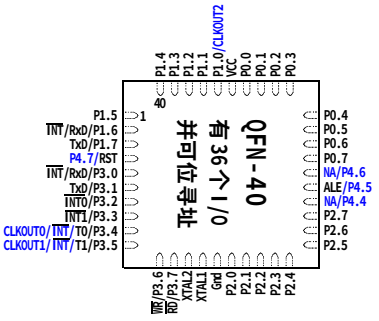
UART_P1: 0， 串口 /UART 在 P3 口[RxD/P3.0，TxD/P3.1]
1， 串口 /UART 在 P1 口，将串口从 P3 口切换到 P1 口[RxD/P1.6,TxD/P1.7]

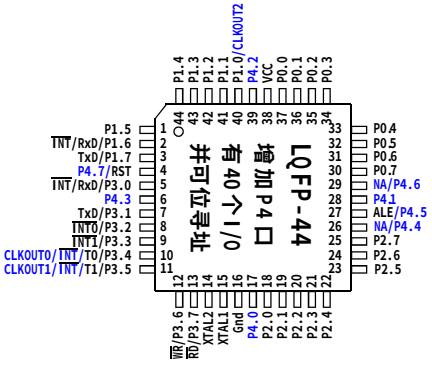
STC10xx 系列单片机管脚排列如下所示：

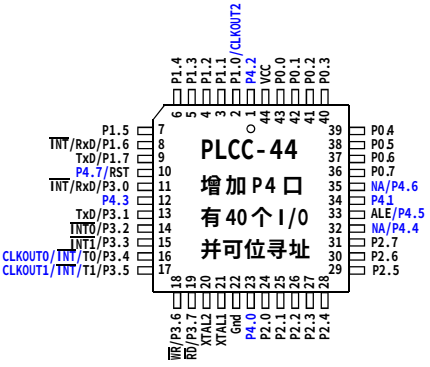
串行口做主机通信时，可控制串口通信在[RxD/P3.0,TxD/P3.1]和[RxD/P1.6,TxD/P1.7.]之间任意切换，实现 2 组串口。建议用户将自己的串行口设置在[RxD/P1.6,TxD/P1.7.]而将[RxD/P3.0,TxD/P3.1]口作为 ISP 下载的专用通信口,当然也可以当用户的普通 I/O 口用

如将复位脚 /RST 当 I/O 口使用，必须使用外部时钟









STC10F08/STC10L08 系列(无内部扩展 256 字节 RAM，无内部 EEPROM)

STC10F08X/STC10L08X 系列(有内部扩展 256 字节 RAM)

STC10F08XE/STC10L08XE 系列(有内部扩展 256 字节 RAM,有内部 EEPROM)

由 P4SW 寄存器设置(NA/P4.4，ALE/P4.5，NA/P4.6)三个端口的第二功能

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
P4SW	BBh	Port-4 switch	-	NA_P4.6	ALE_P4.5	NA_P4.4	-	-	-	-	xOOO,xxxx

NA/P4.4: 0，复位后 P4SW.4 = 0,NA/P4.4 脚是弱上拉，无任何功能

1,通过设置 P4SW.4 = 1,将 NA/P4.4 脚设置成 I/O 口(P4.4)

ALE/P4.5: 0，复位后 P4SW.5 = 0,ALE/P4.5 脚是 ALE 信号,只有在用 MOVX 指令访问片外扩展器件时才有信号输出

1,通过设置 P4SW.5 = 1,将 ALE/P4.5 脚设置成 I/O 口(P4.5)

NA/P4.6: 0，复位后 P4SW.6 = 0,NA/P4.6 脚是弱上拉，无任何功能

1,通过设置 P4SW.6 = 1 将 NA/P4.6 脚设置成 I/O 口(P4.6)

在 ISP 烧录程序时设置 LQFP44/PDIP40/PLCC44 封装的单片机 RST/P4.7 管脚的第二功能，

RST/P4.7 在 ISP 烧录程序时选择是复位脚还是 P4.7 口，如设置成 P4.7 口，必须使用外部时钟。

由 AUXR1 寄存器设置(串口 /UART)是在 P3 口还是在 P1 口

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR1	A2h	Auxiliary register 1	UART_P1	-	-	-	GF2	-	-	DPS	0xxx,0xx0

UART_P1: 0， 串口 /UART 在 P3 口[RxD/P3.0，TxD/P3.1]

1， 串口 /UART 在 P1 口，将串口从 P3 口切换到 P1 口[RxD/P1.6,TxD/P1.7]

GF2: 通用标志位

DPS: 0,使用缺省数据指针 DPTR0

1,使用缺省数据指针 DPTR1

1.3 STC11/10xx系列单片机选型一览表

STC11xx 系列单片机选型一览表

型号	工作电压(V)	Flash程序存储器字节	SRAM字节	EEPROM	定时器T0 T1	UART串口有独立波特率发生器	DPTR	中断优先级	内部低压中断	支持掉电唤醒外部中断	掉电唤醒专用定时器	内置复位并可选择复位门槛电压	看门狗	封装16-Pin 12个I/O	封装18-Pin 14个I/O	封装20-Pin 16个I/O
STC11Fxx系列单片机选型一览																
STC11F01	5.5 - 4.1/3.5	1K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F02	5.5 - 4.1/3.5	2K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F03	5.5 - 4.1/3.5	3K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F04	5.5 - 4.1/3.5	4K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F05	5.5 - 4.1/3.5	5K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
IAP11F06	5.5 - 4.1/3.5	6K	256	-	有	1-2个	1	2	有	5个	有	有	有	可在程序区修改程序区		
STC11F01E	5.5 - 4.1/3.5	1K	256	2K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F02E	5.5 - 4.1/3.5	2K	256	2K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F03E	5.5 - 4.1/3.5	3K	256	2K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F04E	5.5 - 4.1/3.5	4K	256	1K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11F05E	5.5 - 4.1/3.5	5K	256	1K	有	1-2个	1	2	有	5个	有	有	有	需P1.0/P1.1 = 0/0和外部时钟才可以下载用户程序		
STC11Lxx系列单片机选型一览																
型号	工作电压(V)	Flash程序存储器字节	SRAM字节	EEPROM	定时器T0 T1	UART串口有独立波特率发生器	DPTR	中断优先级	内部低压中断	支持掉电唤醒外部中断	掉电唤醒专用定时器	内置复位并可选择复位门槛电压	看门狗	封装16-Pin 12个I/O	封装18-Pin 14个I/O	封装20-Pin 16个I/O
STC11L01	3.6 - 2.4/2.1	1K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L02	3.6 - 2.4/2.1	2K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L03	3.6 - 2.4/2.1	3K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L04	3.6 - 2.4/2.1	4K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L05	3.6 - 2.4/2.1	5K	256	-	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
IAP11L06	3.6 - 2.4/2.1	6K	256	-	有	1-2个	1	2	有	5个	有	有	有	可在程序区修改程序区		
STC11L01E	3.6 - 2.4/2.1	1K	256	2K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L02E	3.6 - 2.4/2.1	2K	256	2K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L03E	3.6 - 2.4/2.1	3K	256	2K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L04E	3.6 - 2.4/2.1	4K	256	1K	有	1-2个	1	2	有	5个	有	有	有	SOP/DIP	SOP/DIP	SOP/DIP
STC11L05E	3.6 - 2.4/2.1	5K	256	1K	有	1-2个	1	2	有	5个	有	有	有	需P1.0/P1.1 = 0/0和外部时钟才可以下载用户程序		

注意事项： STC11xx 和 STC10xx 全系列都有一个独立波特率发生器，STC11xx 和 STC10xx 系列的区别是:STC11xx 比 STC10xx 系列多了一个掉电唤醒专用定时器

STC11F05,STC11F05E,STC11L05,STC11L05E,IAP11F06,IAP11L06 IAP11F62,IAP11F62X,IAP11L62,IAP11L62X 在下载用户程序时，需 将 P1.0/P1.1 短接到地,同时需使用外部时钟才可下载用户程序

STC11Fxx系列单片机选型一览表

型 号	工作电压(V)	Flash程序存储器字节	SRAM字节	EEPROM	定时器T0 T1	UART串口有独立波特率发生器	DPTR	中断优先级	内部低压中断	支持掉电唤醒外部中断	掉电唤醒专用定时器	内置复位并可选择复位门槛电压	看门狗	封装 40-Pin 36个I/O	封装 44-Pin 40个I/O
STC11Fxx系列单片机选型一览															
STC11F60XE	5.5 - 4.1/3.7	60K	1280	1K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F56XE	5.5 - 4.1/3.7	56K	1280	5K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F52XE	5.5 - 4.1/3.7	52K	1280	9K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F48XE	5.5 - 4.1/3.7	48K	1280	13K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F40XE	5.5 - 4.1/3.7	40K	1280	21K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F32XE	5.5 - 4.1/3.7	32K	1280	29K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F20XE	5.5 - 4.1/3.7	20K	1280	29K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F16XE	5.5 - 4.1/3.7	16K	1280	32K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F08XE	5.5 - 4.1/3.7	8K	1280	32K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F60X	5.5 - 4.1/3.7	60K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F56X	5.5 - 4.1/3.7	56K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F52X	5.5 - 4.1/3.7	52K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F48X	5.5 - 4.1/3.7	48K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F40X	5.5 - 4.1/3.7	40K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F32X	5.5 - 4.1/3.7	32K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F20X	5.5 - 4.1/3.7	20K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F16X	5.5 - 4.1/3.7	16K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F08X	5.5 - 4.1/3.7	8K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
IAP11F62X	5.5 - 4.1/3.7	62K	1280		有	1/2	2	2	有	5个	有	有	有	可在程序区修改程序区	
STC11F60	5.5 - 4.1/3.7	60K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F56	5.5 - 4.1/3.7	56K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F52	5.5 - 4.1/3.7	52K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F48	5.5 - 4.1/3.7	48K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F40	5.5 - 4.1/3.7	40K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F32	5.5 - 4.1/3.7	32K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F20	5.5 - 4.1/3.7	20K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F16	5.5 - 4.1/3.7	16K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11F08	5.5 - 4.1/3.7	8K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
IAP11F62	5.5 - 4.1/3.7	62K	256	-	有	1/2	2	2	有	5个	有	有	有	可在程序区修改程序区	

注意事项： STC11xx 和 STC10xx 全系列都有一个独立波特率发生器，STC11xx 和 STC10xx 系列的区别是:STC11xx 比 STC10xx 系列多了一个掉电唤醒专用定时器

STC11F05,STC11F05E,STC11L05,STC11L05E,IAP11F06,IAP11L06 IAP11F62,IAP11F62X,IAP11L62,IAP11L62X 在下载用户程序时，需 将 P1.0/P1.1 短接到地,同时需使用外部时钟才可下载用户程序

STC11Lxx系列单片机选型一览表

型 号	工作电压(V)	Flash程序存储器字节	SRAM字节	EEPROM	定时器T0 T1	UART串口有独立波特率发生器	DPTR	中断优先级	内部低压中断	支持掉电唤醒外部中断	掉电唤醒专用定时器	内置复位并可选择复位门槛电压	看门狗	封装 40-Pin 36个I/O	封装 44-Pin 40个I/O
STC11Fxx系列单片机选型一览															
STC11L60XE	3.6 - 2.4/2.1	60K	1280	1K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L56XE	3.6 - 2.4/2.1	56K	1280	5K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L52XE	3.6 - 2.4/2.1	52K	1280	9K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L48XE	3.6 - 2.4/2.1	48K	1280	13K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L40XE	3.6 - 2.4/2.1	40K	1280	21K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L32XE	3.6 - 2.4/2.1	32K	1280	29K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L20XE	3.6 - 2.4/2.1	20K	1280	29K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L16XE	3.6 - 2.4/2.1	16K	1280	32K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L08XE	3.6 - 2.4/2.1	8K	1280	32K	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L60X	3.6 - 2.4/2.1	60K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L56X	3.6 - 2.4/2.1	56K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L52X	3.6 - 2.4/2.1	52K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L48X	3.6 - 2.4/2.1	48K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L40X	3.6 - 2.4/2.1	40K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L32X	3.6 - 2.4/2.1	32K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L20X	3.6 - 2.4/2.1	20K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L16X	3.6 - 2.4/2.1	16K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L08X	3.6 - 2.4/2.1	8K	1280	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
IAP11L62X	3.6 - 2.4/2.1	62K	1280		有	1/2	2	2	有	5个	有	有	有	可在程序区修改程序区	
STC11L60	3.6 - 2.4/2.1	60K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L56	3.6 - 2.4/2.1	56K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L52	3.6 - 2.4/2.1	52K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L48	3.6 - 2.4/2.1	48K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L40	3.6 - 2.4/2.1	40K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L32	3.6 - 2.4/2.1	32K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L20	3.6 - 2.4/2.1	20K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L16	3.6 - 2.4/2.1	16K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
STC11L08	3.6 - 2.4/2.1	8K	256	-	有	1-2个	2	2	有	5个	有	有	有	PDIP	LQFP/PLCC
IAP11L62	3.6 - 2.4/2.1	62K	256	-	有	1/2	2	2	有	5个	有	有	有	可在程序区修改程序区	

注意事项： STC11xx 和 STC10xx 全系列都有一个独立波特率发生器，STC11xx 和 STC10xx 系列的区别是:STC11xx 比 STC10xx 系列多了一个掉电唤醒专用定时器

STC11F05,STC11F05E,STC11L05,STC11L05E,IAP11F06,IAP11L06 IAP11F62,IAP11F62X,IAP11L62,IAP11L62X 在下载用户程序时，需 将 P1.0/P1.1 短接到地,同时需使用外部时钟才可下载用户程序

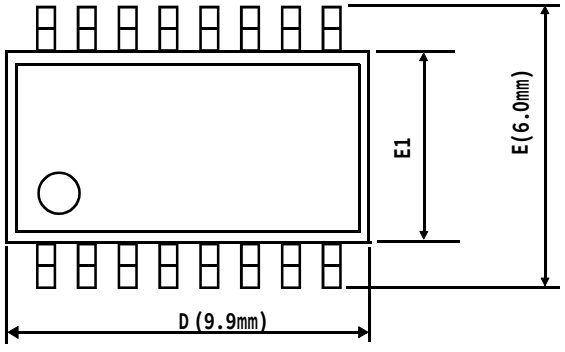
STC10xx系列单片机选型一览表

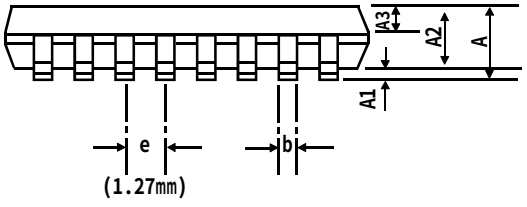
型 号	工作电压(V)	Flash程序存储器字节	SRAM字节	EEPROM	定时器T0 T1	UART串口有独立波特率发生器	DPTR	中断优先级	内部低压中断	支持掉电唤醒外部中断	掉电唤醒专用定时器	内置复位并可选择复位门槛电压	看门狗	封装 40-Pin 36个I/O	封装 44-Pin 40个I/O
STC10Fxx系列单片机选型一览															
STC10F02	5.5 - 3.8/3.3	2K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F02X	5.5 - 3.8/3.3	2K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F02XE	5.5 - 3.8/3.3	2K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F04	5.5 - 3.8/3.3	4K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F04X	5.5 - 3.8/3.3	4K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F04XE	5.5 - 3.8/3.3	4K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F06	5.5 - 3.8/3.3	6K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F06X	5.5 - 3.8/3.3	6K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F06XE	5.5 - 3.8/3.3	6K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F08	5.5 - 3.8/3.3	8K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F08X	5.5 - 3.8/3.3	8K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F08XE	5.5 - 3.8/3.3	8K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F10	5.5 - 3.8/3.3	10K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F10X	5.5 - 3.8/3.3	10K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F10XE	5.5 - 3.8/3.3	10K	512	3K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F12	5.5 - 3.8/3.3	12K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F12X	5.5 - 3.8/3.3	12K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10F12XE	5.5 - 3.8/3.3	12K	512	1K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
IAP10F14	5.5 - 3.8/3.3	14K	512		有	1-2个	2	2	有	5个	-	有	有	可在程序区修改程序区	
STC10Lxx系列单片机选型一览															
型 号	工作电压(V)	Flash程序存储器字节	SRAM字节	EEPROM	定时器T0 T1	UART串口有独立波特率发生器	DPTR	中断优先级	内部低压中断	支持掉电唤醒外部中断	掉电唤醒专用定时器	内置复位并可选择复位门槛电压	看门狗	封装 40-Pin 36个I/O	封装 44-Pin 40个I/O
STC10L02	3.6- 2.4/2.1	2K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L02X	3.6- 2.4/2.1	2K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L02XE	3.6- 2.4/2.1	2K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L04	3.6- 2.4/2.1	4K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L04X	3.6- 2.4/2.1	4K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L04XE	3.6- 2.4/2.1	4K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L06	3.6- 2.4/2.1	6K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L06X	3.6- 2.4/2.1	6K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L06XE	3.6- 2.4/2.1	6K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L08	3.6- 2.4/2.1	8K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L08X	3.6- 2.4/2.1	8K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L08XE	3.6- 2.4/2.1	8K	512	5K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L10	3.6- 2.4/2.1	10K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L10X	3.6- 2.4/2.1	10K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L10XE	3.6- 2.4/2.1	10K	512	3K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L12	3.6- 2.4/2.1	12K	256	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L12X	3.6- 2.4/2.1	12K	512	-	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
STC10L12XE	3.6- 2.4/2.1	12K	512	1K	有	1-2个	2	2	有	5个	-	有	有	PDIP	LQFP/PLCC
IAP10L14	3.6- 2.4/2.1	14K	512		有	1-2个	2	2	有	5个	-	有	有	可在程序区修改程序区	

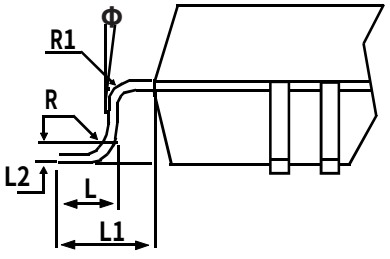
1.4 STC11/10xx系列单片机封装尺寸图

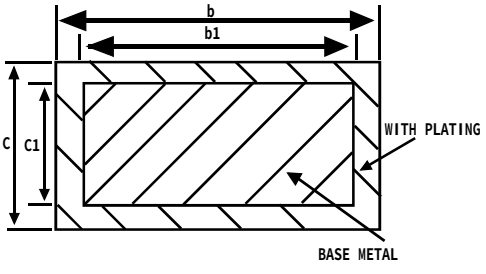
SOP-16封装尺寸图

16-PIN SMALL OUTLINE PACKAGE (SOP-16)







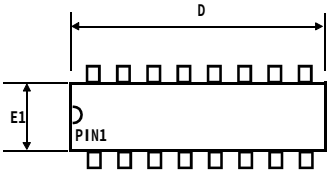


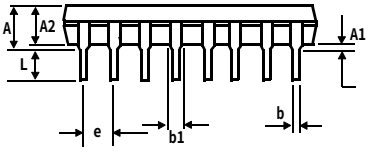
COMMON DIMENSIONS			
(UNITS OF MEASURE = MILLIMETER)			
SYMBOL	MIN	NOM	MAX
A	1.35	1.60	1.75
A1	0.10	0.15	0.25
A2	1.25	1.45	1.65
A3	0.55	0.65	0.75
b	0.36	-	0.49
b1	0.35	0.40	0.45
c	0.16	-	0.25
c1	0.15	0.20	0.25
D	9.80	9.90	10.00
E	5.80	6.00	6.20
E1	3.80	3.90	4.00
e	1.27BSC		
L	0.45	0.60	0.80
L1	1.04REF		
L2	0.25BSC		
R	0.07	-	-
R1	0.07	-	-
Φ	6°	8°	10°

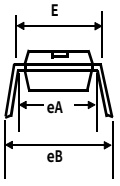
PDIP-16封装尺寸图

Plastic Dual Inline Package (PDIP-16)

Dimensions in Inches and (Millimeters)



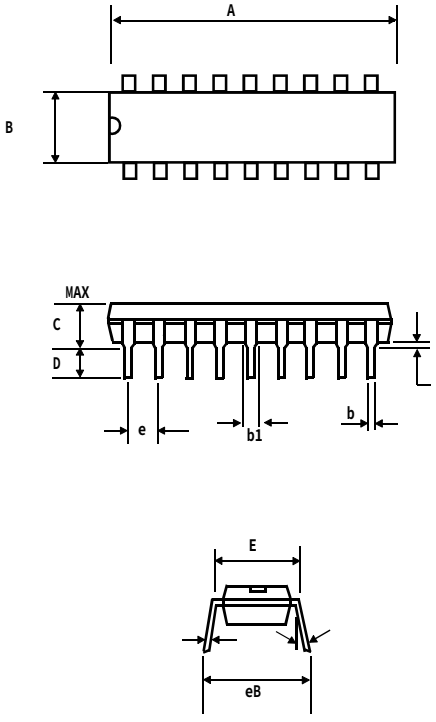




COMMON DIMENSIONS			
(UNITS OF MEASURE = MILLIMETER)			
SYMBOL	MIN	NOM	MAX
A	-	-	4.80
A1	0.50	-	-
A2	3.10	3.30	3.50
b	0.38	-	0.55
b1	0.38	0.46	0.51
D	18.95	19.05	19.15
E	7.62	7.87	8.25
E1	6.25	6.35	6.45
e	2.54BSC		
eA	7.62BSC		
eB	7.62	8.80	10.90
L	2.92	3.30	3.81

PDIP-18 封装尺寸图

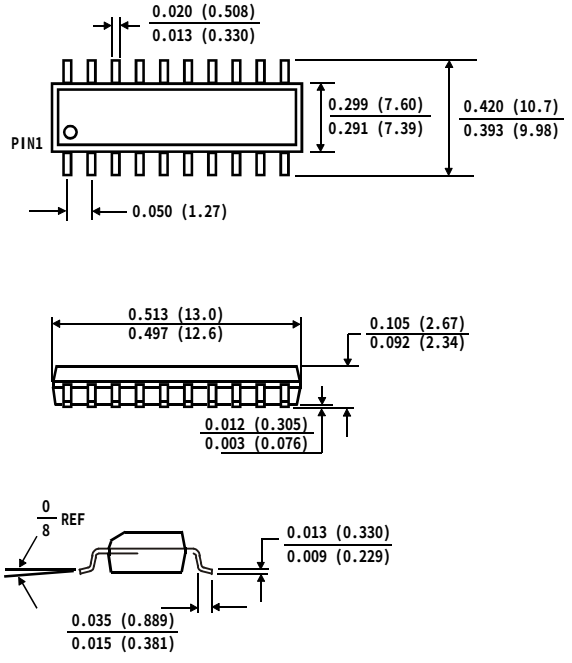
Plastic Dual Inline Package (PDIP-18)
Dimensions in Inches and (Millimeters)



COMMON DIMENSIONS			
(UNITS OF MEASURE = MILLIMETER)			
SYMBOL	MIN	NOM	MAX
A	22.72	-	23.23
B	6.10	-	6.60
C	3.18	-	3.43
D	3.18	-	3.69
e	-	2.54	-
b	0.41	-	0.51
b1	1.27	-	1.78
E	7.49	-	8.00
eB	8.51	-	9.52

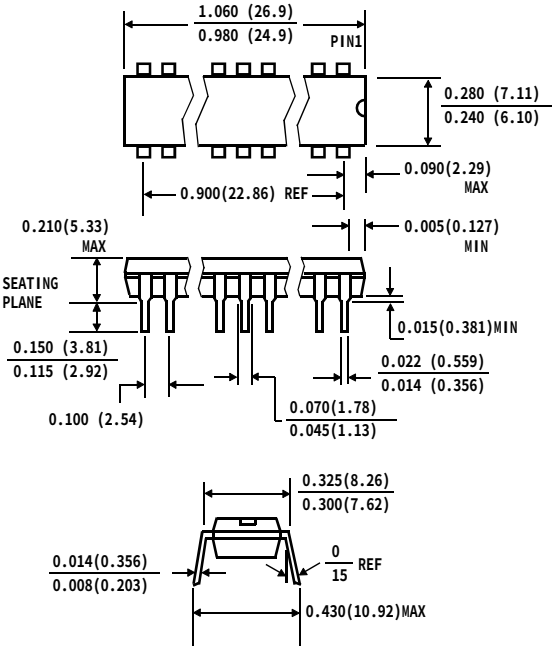
SOP-20封装尺寸图

Plastic Gull WIng Small Outline (SOIC-20 / SOP-20)
Dimensions in Inches and (Millimeters)



PDIP-20封装尺寸图

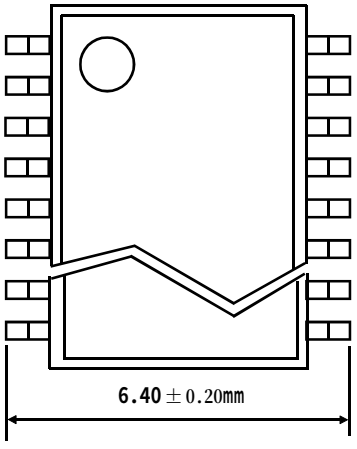
Plastic Dual Inline Package (PDIP-20)
Dimensions in Inches and (Millimeters)

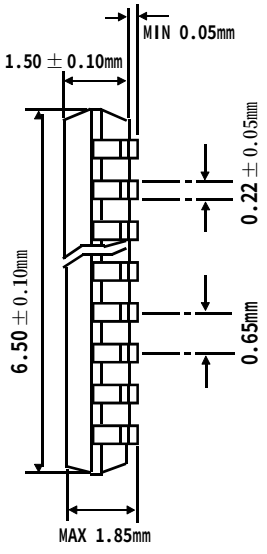


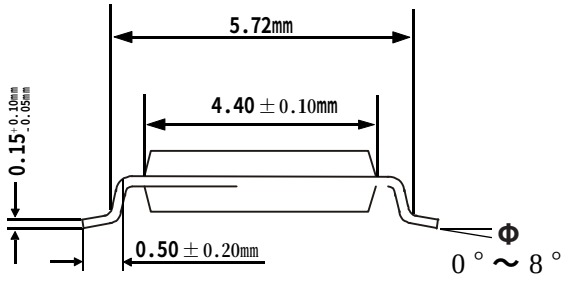
LSSOP-20 封装尺寸图

超小封装 LSSOP-20(仅为 6.4mm x 6.4mm),尺寸只有常规的 SOP-8 大小

PACKAGE：PLASTIC SHRINK SMALL OUTLINE（LSSOP-20，6.4mm × 6.4mm）

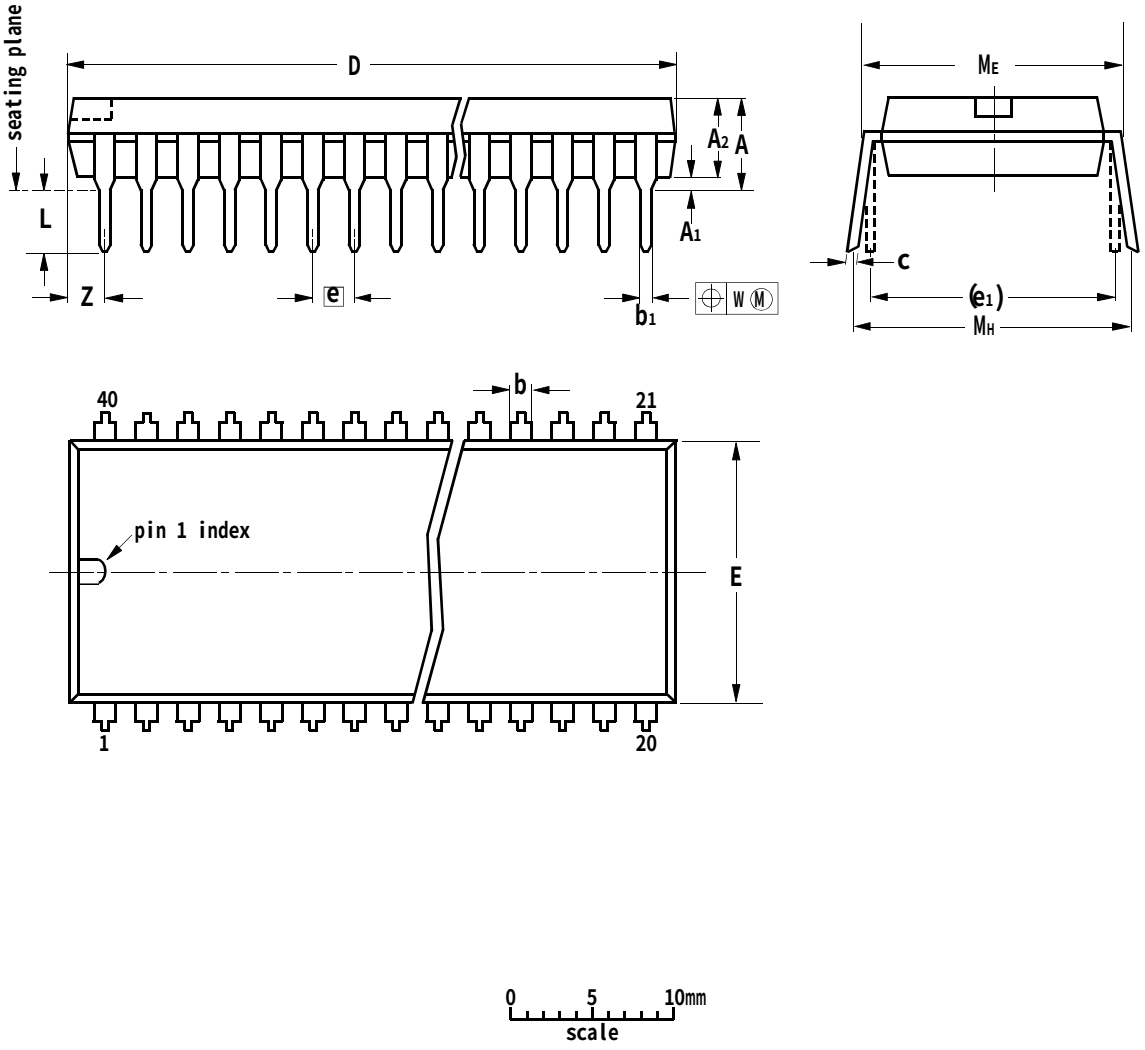






PDIP-40 封装尺寸图

PDIP40: plastic dual in-line package;40 leads(600 mil)



DIMENSIONS(inch dimensions are derived from the original mm dimensions)

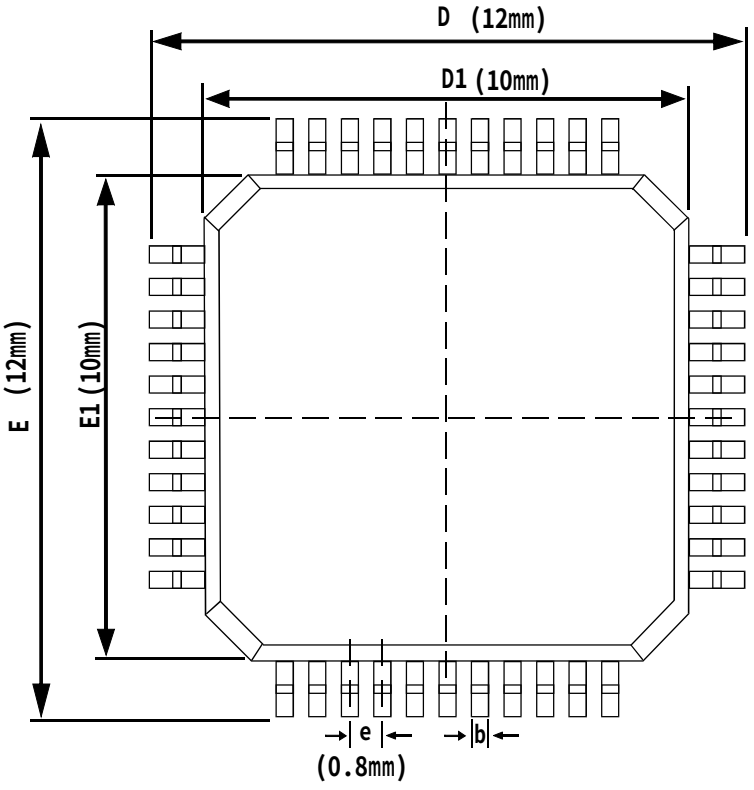
UNIT	A max.	A1 min.	A2 max.	b	b1	c	D(1)	E(1)	e	e1	L	ME	MH	W	Z(1) max.
mm	4.7	0.51	4.0	1.70 1.14	0.53 0.38	0.36 0.23	52.5 51.5	14.1 13.7	2.54	15.24	3.60 3.05	15.8 15.24	17.42 15.90	0.254	2.25
inches	0.19	0.020	0.16	0.067 0.045	0.021 0.015	0.014 0.009	2.067 2.028	0.56 0.54	0.10	0.60	0.14 0.12	0.62 0.60	0.69 0.63	0.01	0.089

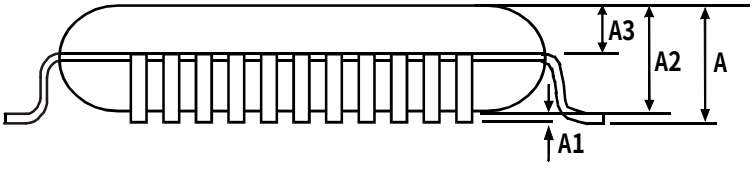
Note
1.Plastic or metal protrusion of 0.25 mm maximum per side are not included

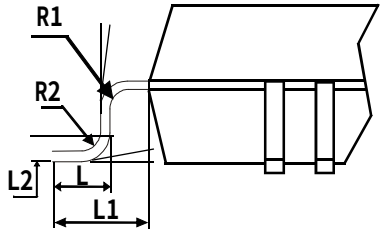
OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	EIAJ			
SOT129-1	051G08	MO-015	SC-511-40			~~95-01-14~~ 99-12-27

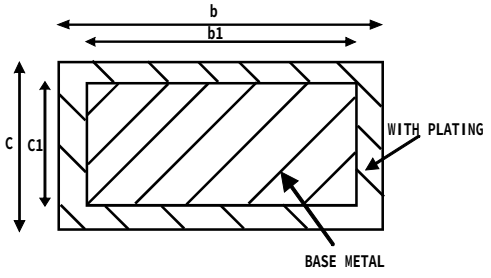
LQFP-44 封装尺寸图

LQFP-44 OUTLINE PACKAGE



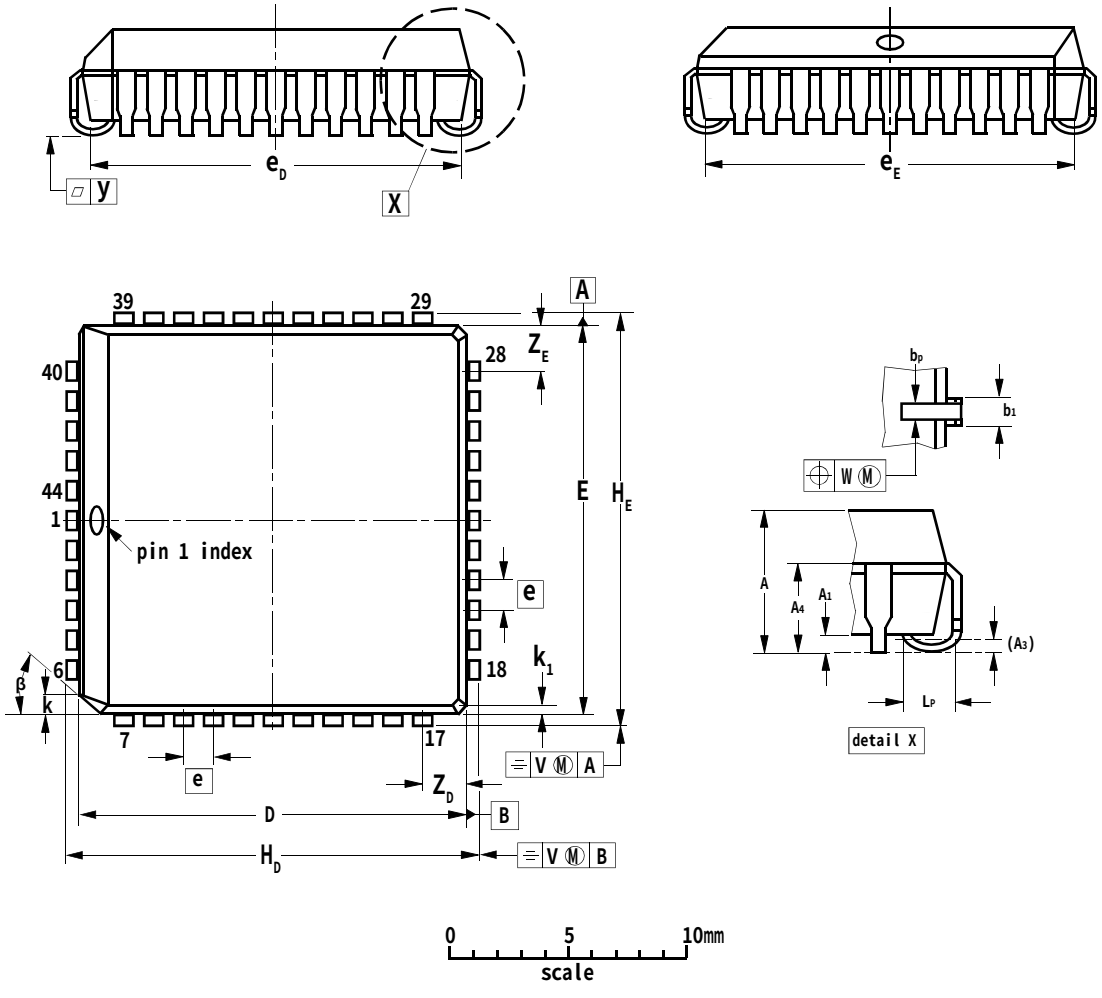






COMMON DIMENSIONS			
(UNITS OF MEASURE = MILLIMETER)			
SYMBOL	MIN	NOM	MAX
A	-	-	1.20
A1	0.05	-	0.15
A2	0.95	1.00	1.05
A3	0.39	0.44	0.49
b	0.31	-	0.44
b1	0.30	0.35	0.40
c	0.13	-	0.18
c1	0.12	0.127	0.134
D	11.80	12.00	12.20
D1	9.90	10.00	10.10
E	11.80	12.00	12.20
E1	9.90	10.00	10.10
e	0.80BSC		
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	-	-
R2	0.08	-	0.20
S	0.20	-	-
Φ	0°	3.5°	7°
Φ1	0°	–	–
Φ2	11°	12°	13°
Φ3	11°	12°	13°

PLCC-44 封装尺寸图



DIMENSIONS(millimetre dimensions are derived from the original inch dimensions)

UNIT	A	A_1 max.	A_3	A_4 max.	b_p	b_1	$D^{(1)}$	$E^{(1)}$	e	e_D	e_E	H_D	H_E	k	k_1 max.	L_p	v	w	y	$Z_D^{(1)}$ max.	$Z_E^{(1)}$ max.	β
mm	4.57 4.19	0.51	0.25	3.05	0.53 0.33	0.81 0.66	16.66 16.51	16.66 16.51	1.27	16.00 14.99	16.00 14.99	17.65 17.40	17.65 17.40	1.22 1.07	0.51	1.44 1.02	0.18	0.18	0.10	2.16	2.16	45 °
inches	0.180 0.165	0.020	0.01	0.12	0.021 0.013	0.032 0.026	0.656 0.650	0.656 0.650	0.05	0.630 0.590	0.630 0.590	0.695 0.685	0.695 0.685	0.048 0.042	0.020	0.057 0.040	0.007	0.007	0.004	0.085	0.085	

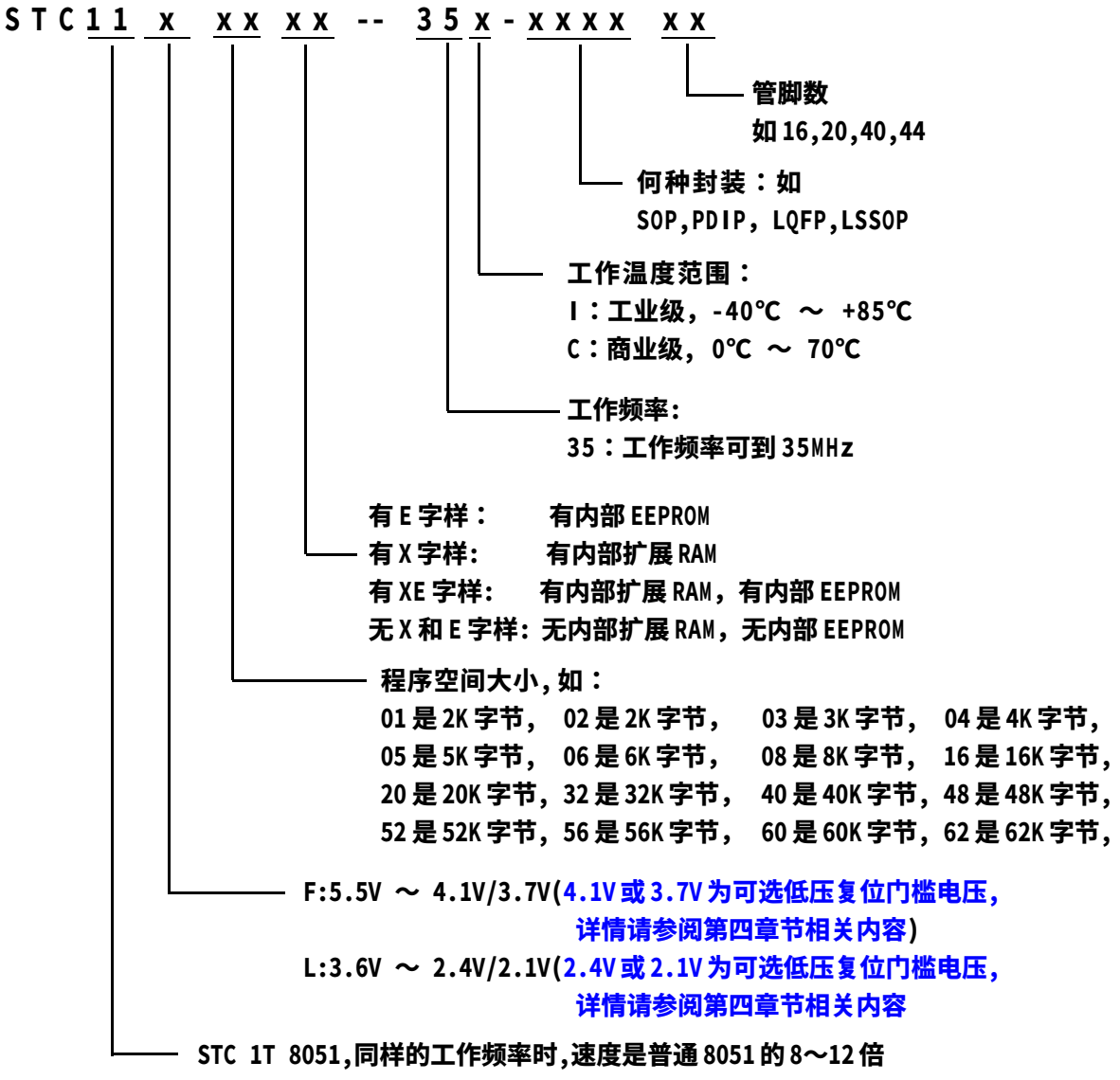
Note

1.Plastic or metal protrusions of 0.01 inches maximum per side are not included

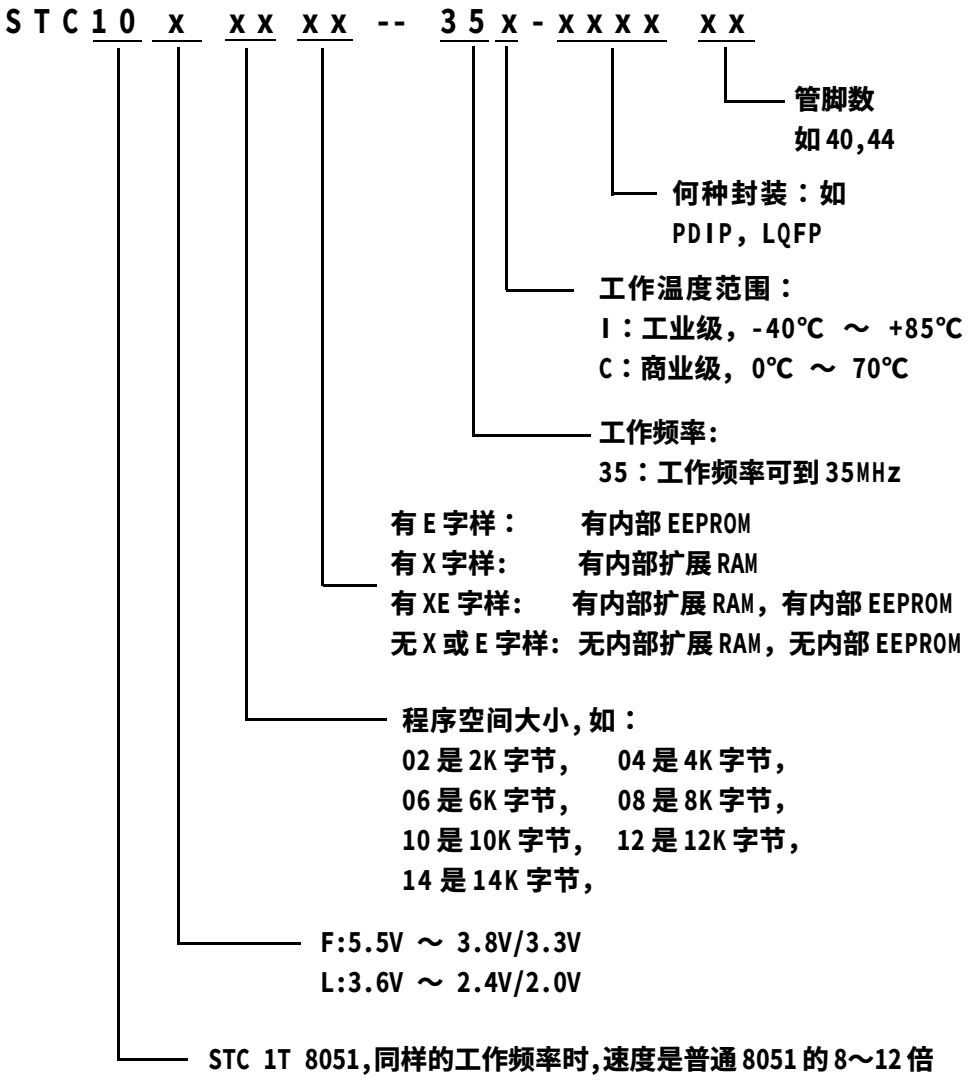
OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	EIAJ			
SOT187-2	112E10	MO-047				~~97-12-16~~ 99-12-27

1.5 STC11/10xx系列单片机命名规则

STC11xx

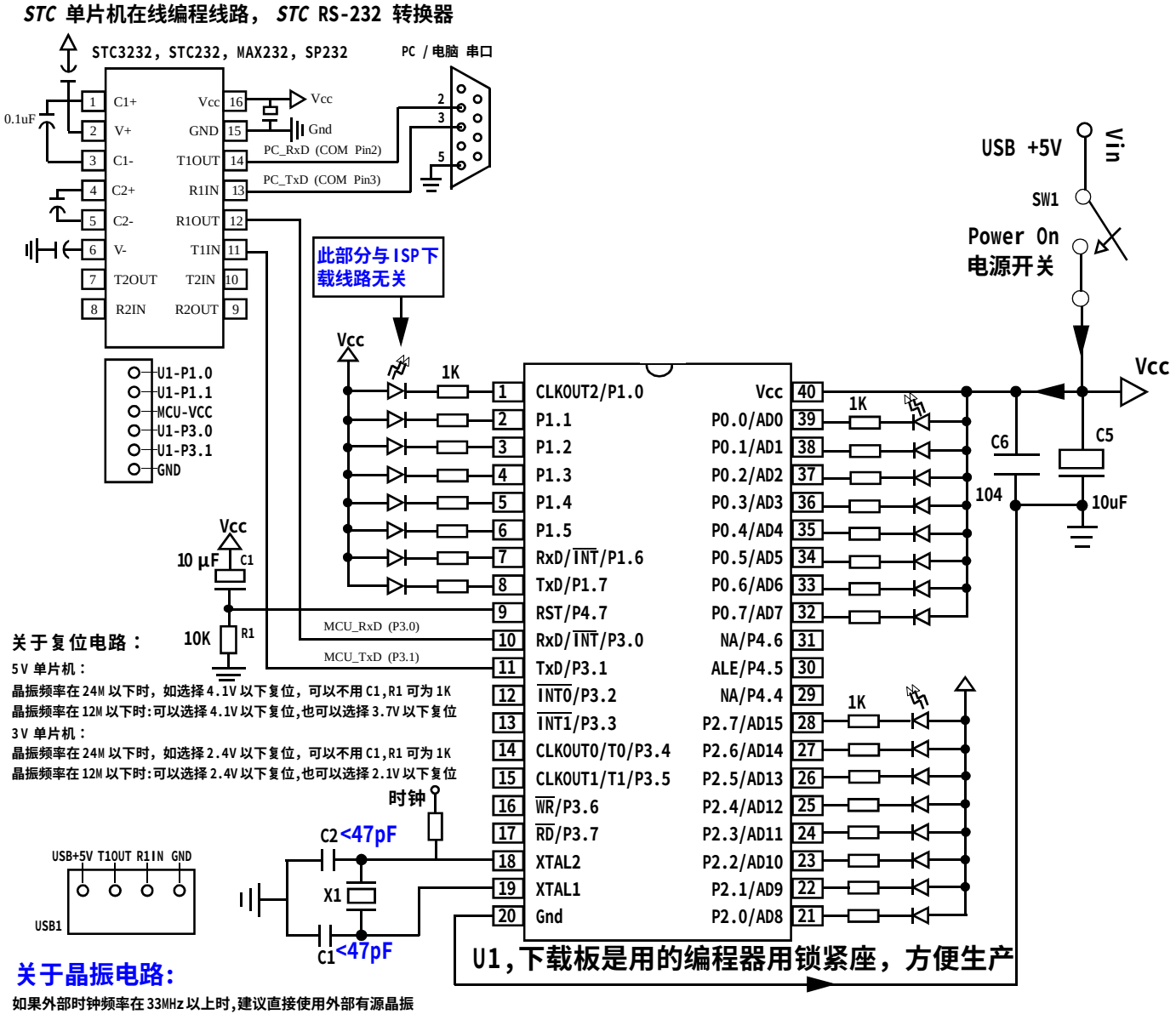


STC10xx 系列单片机命名规则



1.6 STC11/10xx 系列单片机典型应用电路

串行口做主机通信时，可控制串口通信在[RxD/P3.0,TxD/P3.1]和[RxD/P1.6,TxD/P1.7.]之间任意切换，实现 2 组串口。建议用户将自己的串行口设置在[RxD/P1.6,TxD/P1.7.]而将[RxD/P3.0,TxD/P3.1]口作为 ISP 下载的专用通信口,当然也可以当用户的普通 I/O 口用



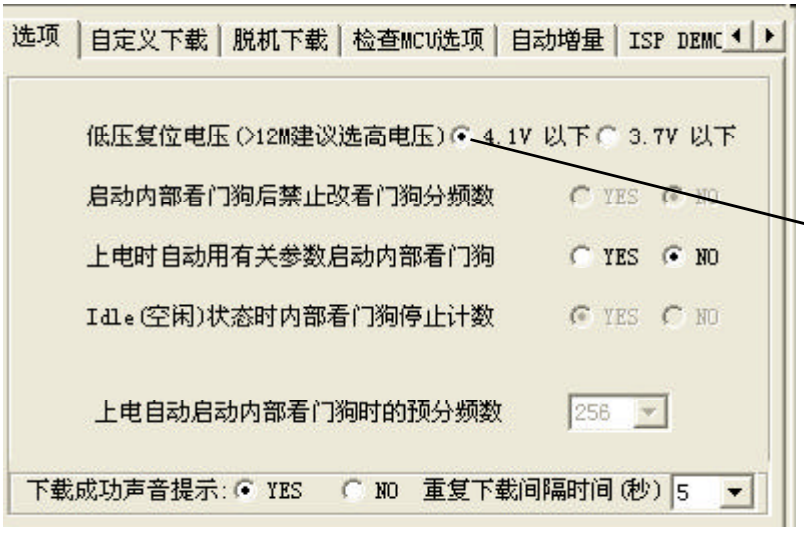
注意事项：

传统 8051 单片机除了在访问片外 64k 数据总线时有 ALE 地址锁存信号输出外,在不访问片外 64k 数据总线时也会输出一个时钟(对系统时钟进行 6 分频输出),此时钟对于不需要的系统来说是一个干扰源.而宏晶最新一代单片机 STC11/10xx 系列基于此原因，将此干扰源彻底切断，将 ALE 本不需要的时钟输出功能拿掉，但继续保留了必要的功能，访问片外 64k 数据总线时 ALE 脚有地址锁存信号输出。大大降低了单片机内部时钟对外部的电磁辐射,提高了系统的可靠性和稳定性,如客户有需要此信号作为其它外围器件的时钟,可以通过如下管脚输出时钟获得:

CLKOUT0/P3.4，CLKOUT1/P3.5，CLKOUT2/P1.0 或者从 XTAL2 脚获取时钟作为其它器件的时钟源(建议在 XTAL2 脚串接一个 200 欧姆的电阻).

1.7 复位门槛电压选择

STC11/10xx 系列单片机都有 2 档复位门槛电压供用户选择



STC11Fxx 系列单片机复位门槛电压选择:

STC11Fxx 系列 5V 单片机:

晶振频率在 24M 以下时，选择 4.1V 以下复位，系统复位才可靠

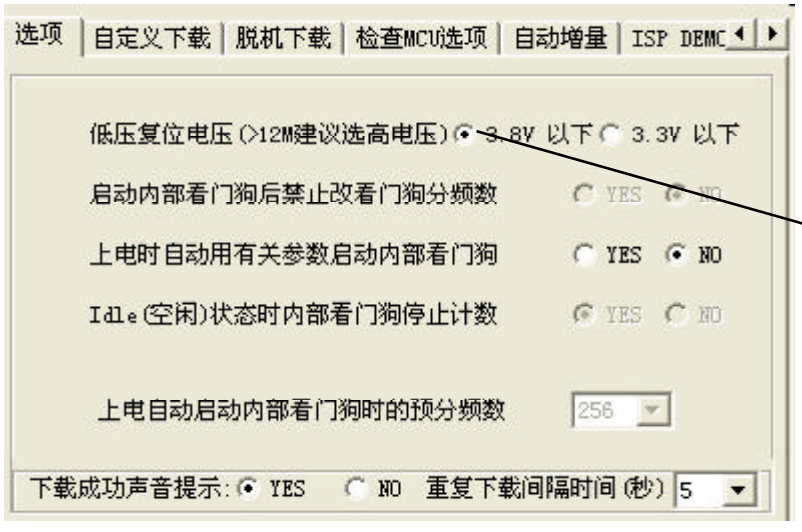
晶振频率在 12M 以下时，可以选择 4.1V 以下复位,也可以选择 3.7V 以下复位

但 STC11F01/02/03/04/05,STC11F01E/02E/03E/04E/05E,IAP11F06 的复位门槛电压为 4.1V 或 3.5V 可选

STC11Lxx 系列 3V 单片机:

晶振频率在 24M 以下时，选择 2.4V 以下复位，系统复位才可靠

晶振频率在 12M 以下时，可以选择 2.4V 以下复位,也可以选择 2.1V 以下复位



STC10Fxx 系列单片机复位门槛电压选择

STC10Fxx 系列 5V 单片机：

晶振频率在 20M 以下时，选择 3.8V 以下复位，系统复位才可靠

晶振频率在 12M 以下时，可以选择 3.8V 以下复位,也可以选择 3.3V 以下复位

STC10Lxx 系列 3V 单片机：

晶振频率在 24M 以下时，选择 2.4V 以下复位，系统复位才可靠

晶振频率在 12M 以下时，可以选择 2.4V 以下复位,也可以选择 2.1V 以下复位

1.8 指令系统分类总结及与普通8051指令执行时间对比

---与8051指令代码完全兼容，但执行的时间效率大幅提升

---其中INC DPTR指令的执行速度大幅提升24倍

---共有12条指令，一个时钟就可以执行完成，平均速度快8～12倍

如果按功能分类，STC11/10xx系列单片机指令系统可分为：

1.数据传送类指令；

2.算术操作类指令；

3.逻辑操作类指令；

4.控制转移类指令；

5.布尔变量操作类指令。

按功能分类的指令系统表如下表所示。

传统 12T 的 8051 指令执行所需时钟

STC11Fxx 系列 指令执行所需时钟

助记符		功能说明	字节数	12时钟/机器周期所需时钟	1时钟/机器周期所需时钟	效率提升
MOV	A，Rn	寄存器内容送入累加器	1	12	1	12倍
MOV	A，direct	直接地址单元中的数据送入累加器	2	12	2	6倍
MOV	A，@Ri	间接RAM中的数据送入累加器	1	12	2	6倍
MOV	A，#data	立即送入累加器	2	12	2	6倍
MOV	Rn，A	累加器内容送入寄存器	1	12	2	6倍
MOV	Rn，direct	直接地址单元中的数据送入寄存器	2	24	4	6倍
MOV	Rn，#data	立即数送入寄存器	2	12	2	6倍
MOV	direct，A	累加器内容送入直接地址单元	2	12	3	4倍
MOV	direct，Rn	寄存器内容送入直接地址单元	2	24	3	8倍
MOV	direct，direct	直接地址单元中的数据送入另一个直接地址单元	3	24	4	6倍
MOV	direct，@Ri	间接RAM中的数据送入直接地址单元	2	24	4	6倍
MOV	direct，#data	立即数送入直接地址单元	3	24	3	8倍
MOV	@Ri，A	累加器内容送间接RAM单元	1	12	3	4倍
MOV	@Ri，direct	直接地址单元数据送入间接RAM单元	2	24	4	6倍
MOV	@Ri，#data	立即数送入间接RAM单元	2	12	3	4倍
MOV	DPTR，#data16	16位立即数送入地址寄存器	3	24	3	8倍
MOVC	A,@A+DPTR	以DPTR为基地址变址寻址单元中的数据送入累加器	1	24	4	6倍
MOVC	A,@A+PC	以PC为基地址变址寻址单元中的数据送入累加器	1	24	4	6倍
MOVX	A,@Ri	逻辑上在外部的片内扩展RAM，(8位地址）送入累加器	1	24	3	8倍
MOVX	A,@DPTR	逻辑上在外部的片内扩展RAM，(16位地址）送入累加器	1	24	3	8倍
MOVX	@Ri,A	累加器送逻辑上在外部的片内扩展RAM（8位地址）	1	24	3	8倍
MOVX	@DPTR ,A	累加器送逻辑上在外部的片内扩展RAM（16位地址）	1	24	3	8倍
MOVX	A,@Ri	物理上在外部的片外扩展RAM，(8位地址）送入累加器	1	24	7	*Note1
MOVX	A,@DPTR	物理上在外部的片外扩展RAM，(16位地址）送入累加器	1	24	7	*Note1
MOVX	@Ri,A	累加器送物理上在外部的片外扩展RAM，（8位地址）	1	24	7	*Note1
MOVX	@DPTR ,A	累加器送物理上在外部的片外扩展RAM，（16位地址）	1	24	7	*Note1
PUSH	direct	直接地址单元中的数据压入堆栈	2	24	4	6倍
POP	direct	出栈送直接地址单元	2	24	3	8倍
XCH	A,Rn	寄存器与累加器交换	1	12	3	4倍
XCH	A，direct	直接地址单元与累加器交换	2	12	4	3倍
XCH	A,@Ri	间接RAM与累加器交换	1	12	4	3倍
XCHD	A,@Ri	间接RAM的低半字节与累加器交换	1	12	4	3倍

Note1: 访问物理上在片外的扩展 RAM 所需时钟：7 + 2 x ALE_Bus_Speed + RW_Bus_Speed

其中 ALE_Bus_Speed 由 BUS_SPEED 控制寄存器中的 ALES1/ALES0 决定

其中 RW_Bus_Speed 由 BUS_SPEED 控制寄存器中的 RWS2/RWS1/RWS0 决定

算术操作类指令

助记符		功能说明	字节数	12时钟/周期所需时钟	1时钟/周期所需时钟	提升效率
ADD	A，Rn	寄存器内容加到累加器	1	12	2	6倍
ADD	A，direct	直接地址单元中的数据加到累加器	2	12	3	4倍
ADD	A，@Ri	间接RAM中的数据加到累加器	1	12	3	4倍
ADD	A，#data	立即加到累加器	2	12	2	6倍
ADDC	A，Rn	寄存器内容带进位加到累加器	1	12	2	6倍
ADDC	A，direct	直接地址单元的内容带进位加到累加器	2	12	3	4倍
ADDC	A，@Ri	间接RAM内容带进位加到累加器	1	12	3	4倍
ADDC	A，#data	立即数带进位加到累加器	2	12	2	6倍
SUBB	A，Rn	累加器带借位减寄存器内容	1	12	2	6倍
SUBB	A，direct	累加器带借位减直接地址单元的内容	2	12	3	4倍
SUBB	A，@Ri	累加器带借位减间接RAM中的内容	1	12	3	4倍
SUBB	A，#data	累加器带借位减立即数	2	12	2	6倍
INC	A	累加器加1	1	12	2	6倍
INC	Rn	寄存器加1	1	12	3	4倍
INC	direct	直接地址单元加1	2	12	4	3倍
INC	@Ri	间接RAM单元加1	1	12	4	3倍
DEC	A	累加器减1	1	12	2	6倍
DEC	Rn	寄存器减1	1	12	3	4倍
DEC	direct	直接地址单元减1	2	12	4	3倍
DEC	@Ri	间接RAM单元减1	1	12	4	3倍
INC	DPTR	地址寄存器DPTR加1	1	24	1	24倍
MUL	AB	A乘以B	1	48	4	12倍
DIV	AB	A除以B	1	48	5	9.6倍
DA	A	累加器十进制调整	1	12	4	3倍

逻辑操作类指令

助记符		功能说明	字节数	12时钟/周期所需时钟	1时钟/周期所需时钟	提升效率
ANL	A，Rn	累加器与寄存器相“与”	1	12	2	6倍
ANL	A，direct	累加器与直接地址单元相“与”	2	12	3	4倍
ANL	A，@Ri	累加器与间接RAM单元相“与”	1	12	3	4倍
ANL	A，#data	累加器与立即数相“与”	2	12	2	6倍
ANL	direct，A	直接地址单元与累加器相“与”	2	12	4	3倍
ANL	direct，#data	直接地址单元与立即数相“与”	3	24	4	6倍
ORL	A，Rn	累加器与寄存器相“或”	1	12	2	6倍
ORL	A，direct	累加器与直接地址单元相“或”	2	12	3	4倍
ORL	A，@Ri	累加器与间接RAM单元相“或”	1	12	3	4倍
ORL	A，#data	累加器与立即数相“或”	2	12	2	6倍
ORL	direct，A	直接地址单元与累加器相“或”	2	12	4	3倍
ORL	direct，#data	直接地址单元与立即数相“或”	3	24	4	6倍
XRL	A，Rn	累加器与寄存器相“异或”	1	12	2	6倍
XRL	A，direct	累加器与直接地址单元相“异或”	2	12	3	4倍
XRL	A，@Ri	累加器与间接RAM单元相“异或”	1	12	3	4倍
XRL	A，#data	累加器与立即数相“异或”	2	12	2	6倍
XRL	direct，A	直接地址单元与累加器相“异或”	2	12	4	3倍
XRL	direct，#data	直接地址单元与立即数相“异或”	3	24	4	6倍
CLR	A	累加器清“0”	1	12	1	12倍
CPL	A	累加器求反	1	12	2	6倍
RL	A	累加器循环左移	1	12	1	12倍
RLC	A	累加器带进位位循环左移	1	12	1	12倍
RR	A	累加器循环右移	1	12	1	12倍
RRC	A	累加器带进位位循环右移	1	12	1	12倍
SWAP	A	累加器半字节交换	1	12	1	12倍

控制转移类指令

助记符	功能说明	字节数	12时钟/周期所需时钟	1时钟/周期所需时钟	提升效率
ACALL addr11	绝对（短）调用子程序	2	24	6	4倍
LCALL addr16	长调用子程序	3	24	6	4倍
RET	子程序返回	1	24	4	6倍
RETI	中断返回	1	24	4	6倍
AJMP addr11	绝对（短）转移	2	24	3	8倍
LJMP addr16	长转移	3	24	4	6倍
SJMP re1	相对转移	2	24	3	8倍
JMP @A+DPTR	相对于DPTR的间接转移	1	24	3	8倍
JZ re1	累加器为零转移	2	24	3	8倍
JNZ re1	累加器非零转移	2	24	3	8倍
CJNE A，direct，re1	累加器与直接地址单元比较，不相等则转移	3	24	5	4.8倍
CJNE A，#data，re1	累加器与立即数比较，不相等则转移	3	24	4	6倍
CJNE Rn，#data，re1	寄存器与立即数比较，不相等则转移	3	24	4	6倍
CJNE @Ri，#data，re1	间接RAM单元与立即数比较，不相等则转移	3	24	5	4.8倍
DJNZ Rn，re1	寄存器减1，非零转移	3	24	4	6倍
DJNZ direct，re1	直接地址单元减1，非零转移	3	24	5	4.8倍
NOP	空操作	1	12	1	12倍

布尔变量操作类指令					
助记符	功能说明	字节数	12时钟/周期所需时钟	1时钟/周期所需时钟	提升效率
CLR C	清0进位位	1	12	1	12倍
CLR bit	清0直接地址位	2	12	4	3倍
SETB C	置1进位位	1	12	1	12倍
SETB bit	置1直接地址位	2	12	4	3倍
CPL C	进位位求反	1	12	1	12倍
CPL bit	直接地址位求反	2	12	4	3倍
ANL C,bit	进位位和直接地址位相“与”	2	24	3	8倍
ANL C,bit	进位位和直接地址位的反码相“与”	2	24	3	8倍
ORL C,bit	进位位和直接地址位相“或”	2	24	3	8倍
ORL C,bit	进位位和直接地址位的反码相“或”	2	24	3	8倍
MOV C,bit	直接地址位送入进位位	2	12	3	4倍
MOV bit,C	进位位送入直接地址位	2	24	4	6倍
JC rel	进位位为1则转移	2	24	3	8倍
JNC rel	进位位为0则转移	2	24	3	8倍
JB bit,rel	直接地址位为1则转移	3	24	4	6倍
JNB bit,rel	直接地址位为0则转移	3	24	4	6倍
JBC bit,rel	直接地址位为1则转移，该位清0	3	24	5	4.8倍

指令执行速度效率提升总结：

指令系统共包括 111 条指令，其中：

执行速度快 24 倍的	共 1 条
执行速度快 12 倍的	共 12 条
执行速度快 9.6 倍的	共 1 条
执行速度快 8 倍的	共 19 条
执行速度快 6 倍的	共 39 条
执行速度快 4.8 倍的	共 4 条
执行速度快 4 倍的	共 21 条
执行速度快 3 倍的	共 14 条

根据对指令的使用频率分析统计，STC11F/10Fxx 系列 1T 的 8051 单片机比普通的 8051 单片机在同样的工作频率
下运行速度提升了 8～12 倍。

指令执行时钟数统计（供参考）：

指令系统共包括 111 条指令，其中：

1 个时钟就可执行完成的指令	共 12 条
2 个时钟就可执行完成的指令	共 20 条
3 个时钟就可执行完成的指令	共 38 条
4 个时钟就可执行完成的指令	共 34 条
5 个时钟就可执行完成的指令	共 5 条
6 个时钟就可执行完成的指令	共 2 条

1.9 特殊功能寄存器映像 SFR Mapping

	Bit Addressable 可位操作	Non Bit Addressable 不可以位操作 (寄存器地址不能够被8整除的不可以进行位操作)							
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
F8h									FFh
F0h	B 0000,0000								F7h
E8h									EFh
E0h	ACC 0000,0000								E7h
D8h									DFh
D0h	PSW 0000,0000								D7h
C8h									CFh
C0h	P4 1111,1111	WDT_CONTR xx00,0000	IAP_DATA 1111,1111	IAP_ADDRH 0000,0000	IAP_ADDRL 0000,0000	IAP_CMD xxxx,xx00	IAP_TRIG xxxx,xxxx	IAP_CONTR 0000,1000	C7h
B8h	IP x0x0,0000	SADEN		P4SW x000,xxxx					BFh
B0h	P3 1x11,1111	P3M1 0000,0000	P3M0 0000,0000	P4M1 0000,0000	P4M0 0000,0000				B7h
A8h	IE 00x0,0000	SADDR	WKTCL 0000,0000	WKTCH 0xxx,0000					AFh
A0h	P2 1111,1111	BUS_SPEED xx10,x011	AUXR1 xxxx,0xx0					Don't use	A7h
98h	SCON 0000,0000	SBUF xxxx,xxxx			BRT 0000,0000				9Fh
90h	P1 1111,1111	P1M1 0000,0000	P1M0 0000,0000	P0M1 0000,0000	P0M0 0000,0000	P2M1 0000,0000	P2M0 0000,0000	CLK_DIV xxxx,x000	97h
88h	TCON 0000,0000	TMOD 0000,0000	TL0 0000,0000	TL1 0000,0000	TH0 0000,0000	TH1 0000,0000	AUXR 0000,x000	WAKE_CLKO x000,x000	8Fh
80h	P0 xxxx,1111	SP 0000,0111	DPL 0000,0000	DPH 0000,0000				PCON 0011,0000	87h
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	

注意：寄存器地址能够被 8 整除的才可以进行位操作，不能够被 8 整除的不可以进行位操作

特别标出部分为在 Intel 8052 基础上新增加的特殊功能寄存器,一般用户可不管

新增特殊功能寄存器如何声明地址，举例如下：

汇编语言(新增 P4 口地址声明) ： P4 EQU 0C0H

C 语言 (新增 P4 口地址声明) ： sfr P4 = 0xC0

```
sbit  P40 = 0xC0;
sbit  P41 = 0xC1;
sbit  P42 = 0xC2;
```

STC11/10xx 系列 8051 单片机内核特殊功能寄存器 C51 Core SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
ACC	E0h	Accumulator									0000,0000
B	F0h	B Register									0000,0000
PSW	D0h	Program Status Word	CY	AC	F0	RS1	RS0	OV	F1	P	0000,0000
SP	81h	Stack Pointer									0000,0111
DPL	82h	Data Pointer Low Byte									0000,0000
DPH	83h	Data Pointer High Byte									0000,0000

STC11/10xx 系列 8051 单片机系统管理特殊功能寄存器 System Management SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset value
PCON	87h	Power Control	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	000[0],x000
CLK_DIV	97h	Clock Divder	-	-	-	-	-	CLKS2	CLKS1	CLKS0	xxxx,x000

STC11/10xx 系列 8051 单片机 I/O 口 特殊功能寄存器 Port SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
P0	80h	8-bit Port 0	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	1111,1111
P0M1	93h										0000,0000
P0M0	94h										0000,0000
P1	90h	8-bit Port 1	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0	1111,1111
P1M1	91h										0000,0000
P1M0	92h										0000,0000
P2	A0h	8-bit Port 2	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	1111,1111
P2M1	95h										0000,0000
P2M0	96h										0000,0000
P3	B0h	8-bit Port 3	P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0	1111,1111
P3M1	B1h										0000,0000
P3M0	B2h										0000,0000
P4	C0h	8-bit Port 4	P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0	0000,0000
P4M1	B3h										0000,0000
P4M0	B4h										0000,0000
P4SW	BBh	Port-4 switch	-	NA_P4.6	ALE_P4.5	NA_P4.4	-	-	-	-	x000,xxxx
AUXR1	A2h		UART_P1	-	-	-	GF2	-	-	DPS	0xxx,0xx0

STC11/10xx 系列 8051 单片机 定时器 特殊功能寄存器 Timer SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
TCON	88h	Timer / Counter 0 and 1 Control	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
TMOD	89h	Timer / Counter 0 and 1 Modes	GATE GATE1	C/T# C/T1#	M1 M1_1	M0 M1_0	GATE GATE0	C/T# C/T0#	M1 M0_1	M0 M0_0	0000,0000
TL0	8Ah	Timer / Counter 0 Low Byte									0000,0000
TH0	8Ch	Timer / Counter 0 High Byte									0000,0000
TL1	8Bh	Timer / Counter 1 Low Byte									0000,0000
TH1	8Dh	Timer / Counter 1 High Byte									0000,0000
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	000[0],x000

STC11/10xx 系列 8051 单片机 串行口 特殊功能寄存器 Serial I/O Port SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
SCON	98h	Serial Control	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
SBUF	99h	Serial Data Buffer									xxxx,xxxx
SADEN	B9h	Slave Address Mask									0000,0000
SADDR	A9h	Slave Address									0000,0000
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	000,x000
AUXR1	A2h		UART_P1	-	-	-	GF2	-	-	DPS	0xxx,0xx0

STC11/10xx 系列 8051 单片机 看门狗定时器 特殊功能寄存器 Watch Dog Timer SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WDT_CONTR	C1h	Watch-Dog-Timer Control register	WDT_FLAG	-	EN_WDT	CLR_WDT	IDLE_WDT	PS2	PS1	PS0	xx00,0000

STC11/10xx 系列 1T 8051 单片机 中断 特殊功能寄存器 Interrupt SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
IE	A8h	Interrupt Enable	EA	ELVD	-	ES	ET1	EX1	ET0	EX0	00x0,0000
IP	B8h	Interrupt Priority Low	-	PLVD	-	PS	PT1	PX1	PT0	PX0	x0x0,0000
TCON	88h	Timer / Counter 0 and 1 Control	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
SCON	98h	Serial Control	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	000,x000
PCON	87h	Power Control	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
WAKE_CLKO	8Fh	CLK_Output Powerdown_Wakeup Control Register	-	RXD_PIN_IE	T1_PIN_IE	T0_PIN_IE	-	BRTCLKO	T1CLKO	T0CLKO	x000,x000

STC11/10xx 系列 8051 单片机 ISP/IAP 特殊功能寄存器　　ISP/IAP SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
IAP_DATA	C2h	ISP/IAP Flash Data Register									1111,1111
IAP_ADDRH	C3h	ISP/IAP Flash Address High									0000,0000
IAP_ADDRL	C4h	ISP/IAP Flash Address Low									0000,0000
IAP_CMD	C5h	ISP/IAP Flash Command Register	-	-	-	-	-	-	MS1	MS0	xxxx,x000
IAP_TRIG	C6h	ISP/IAP Flash Command Trigger									xxxx,xxxx
IAP_CONTR	C7h	ISP/IAP Control Register	IAPEN	SWBS	SWRST	CMD_FAIL	-	WT2	WT1	WT0	0000,x000

STC11/10xx 系列 8051 单片机 时钟输出和掉电唤醒寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WAKE_CLKO	8Fh	Clk_Output Powerdown_Wakeup Control register	-	RXD_PIN_IE	T1_PIN_IE	T0_PIN_IE	-	BRTCLKO	T1CLKO	T0CLKO	x000,x000

STC11/10xx 系列单片机总线控制特殊功能寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
BUS_SPEED	A1h	Bus-Speed Control	-	-	ALES1	ALES0	-	RWS2	RWS1	RWS0	xx10,x011

1.10 中断优先级及中断寄存器

1.10 中断优先级

STC11/10xx 系列单片机 2 级中断优先级及中断查询次序，与 8051 完全兼容

Interrupt Source 中断源	Vector Address 中断向量地址	Polling Sequence 中断查询次序	中断优先级设置(IP)	优先级0最低	优先级1最高	Interrupt Request 中断请求标志位	Interrupt Enable Control Bit 中断允许控制位
/INT0	0003H	0(最优先)	PX0	0	1	IE0	EX0 / EA
Timer 0	000BH	1	PT0	0	1	TF0	ET0 / EA
/INT1	0013H	2	PX1	0	1	IE1	EX1 / EA
Timer 1	001BH	3	PT1	0	1	TF1	ET1 / EA
UART	0023H	4	PS	0	1	RI + TI	ES / EA
N/A(不用)	002BH	5					
LVD	0033H	6	PLVD	0	1	LVDF	ELVD / EA

通过设置设置 IP，那么中断优先级就有两级，与传统 8051 单片机两级中断优先级完全兼容。

如果使用 C 语言编程，中断查询次序号就是中断号，例如：

```
void  Int0_Routine(void) interrupt   0;
void  Timer0_Routine(void) interrupt 1;
void  Int1_Routine(void) interrupt   2;
void  Timer1_Routine(void) interrupt 3;
void  UART_Routine(void) interrupt   4;
void  LVD_Routine(void) interrupt    6;
```

STC11/10xx 系列 1T 8051 单片机 中断 特殊功能寄存器 Interrupt SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
IE	A8h	Interrupt Enable	EA	ELVD	-	ES	ET1	EX1	ET0	EX0	00x0,0000
IP	B8h	Interrupt Priority Low	-	PLVD	-	PS	PT1	PX1	PT0	PX0	x0x0,0000
TCON	88h	Timer / Counter 0 and 1 Control	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
SCON	98h	Serial Control	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000
PCON	87h	Power Control	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
WAKE_CLKO	8Fh	CLK_Output Powerdown_Wakeup Control Register	-	RXD_PIN_IE	T1_PIN_IE	T0_PIN_IE	-	BRTCLKO	T1CLKO	T0CLKO	x000,x000

1.11 定时器 0/ 定时器 1，UART 串口的速度

STC11/10xx 系列单片机的 AUXR 寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000

定时器 0 和定时器 1：

STC11/10xx 系列是 1T 的 8051 单片机，为了兼容传统 8051，定时器 0 和定时器 1 复位后是传统 8051 的速度，即 12 分频，这是为了兼容传统 8051。但也可不进行 12 分频，实现真正的 1T。

T0x12： 0，定时器 0 是传统 8051 速度，12 分频；1，定时器 0 的速度是传统 8051 的 12 倍，不分频

T1x12： 0，定时器 1 是传统 8051 速度，12 分频；1，定时器 1 的速度是传统 8051 的 12 倍，不分频

如果 UART 串口用定时器 1 做波特率发生器，T1x12 位就可以控制 UART 串口是 12T 还是 1T 了。

UART 串口的模式 0：

STC11/10xx 系列是 1T 的 8051 单片机，为了兼容传统 8051，UART 串口复位后是兼容传统 8051 的。

UART_M0x6： 0，UART 串口的模式 0 是传统 12T 的 8051 速度，12 分频；

1，UART 串口的模式 0 的速度是传统 12T 的 8051 的 6 倍，2 分频

如果用定时器 T1 做波特率发生器时，UART 串口的速度由 T1 的溢出率决定

STC11/10xx 系列单片机的 AUXR 寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000

BRTR： 0， 不允许独立波特率发生器运行

1， 允许独立波特率发生器运行

BRTx12： 0， 独立波特率发生器每 12 个时钟计数一次

1， 独立波特率发生器每 1 个时钟计数一次

XRAM： 0， 允许使用内部扩展的 1024 字节扩展 RAM

1， 禁止使用内部扩展的 1024 字节扩展 RAM

S1BRS： 0， 缺省，串口 1 波特率发生器选择定时器 1，S1BRS 是串口 1 波特率发生器选择位

1， 独立波特率发生器作为串口 1 的波特率发生器，此时定时器 1 得到释放，可以作为独立定时器使用

注 意 ：

串口 1 可以选择定时器 1 做波特率发生器，也可以选择独立波特率发生器作为波特率发生器，

1.12 STC11/10xx系列单片机内部/外部工作时钟可选

STC11/10xx 系列是 1T 的 8051 单片机，系统时钟源可选内部 RC 振荡器,外部晶体或者时钟。

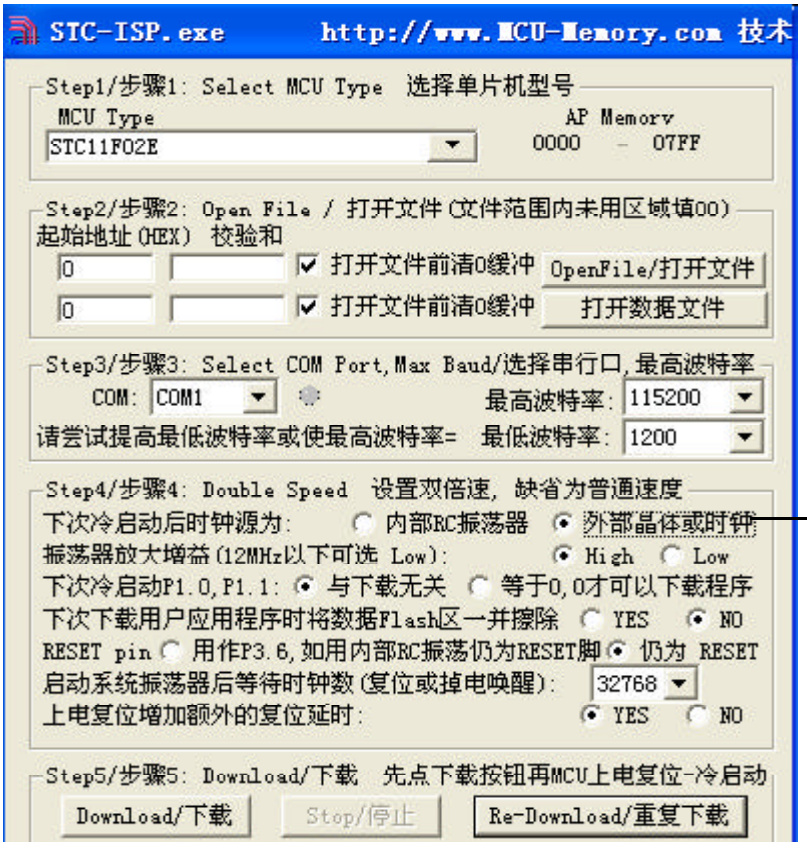
现出厂标准配置是使用芯片内部的 R/C 振荡器，5V 单片机常温下频率是 4MHz - 8MHz,3V 单片机常温下频率是 4MHz - 8MHz,因为随着温度的变化，内部 R/C 振荡器的频率会有一些温飘，再加上制造误差，故内部 R/C 振荡器只适用于对时钟频率要求不敏感的场合。

在对 STC11/10xx 系列单片机进行 ISP 下载用户程序时，可以在选项中选择：

“下次冷启动后时钟源为外部晶体或时钟”

这样下载完用户程序后，停电，再冷启动后单片机的工作时钟使用的就不是内部 R/C 振荡器，而是外部晶体振荡后产生的高精度时钟了（接在 XTAL1/XTAL2 管脚上），也可以直接从 XTAL1 脚输入外部时钟，XTAL2 脚浮空。用户以后外部必须接晶体或时钟单片机才可以工作。

如果已被设置成用外部晶体或时钟工作的单片机，还要再设回使用内部 R/C 振荡器工作，则需给单片机外接晶体或时钟，再对 STC11/10xx 系列单片机进行 ISP 下载用户程序时在选项中选择：



选择下次冷启动后时钟源为：

1.内部 R/C 振荡器

2.外部晶体或时钟

下载用户程序成功后，新的设置就设置进单片机内部了，但必须停电后再上电单片机才会用新的设置工作

1.13 时钟分频及分频寄存器

时钟分频寄存器，可将时钟分成较低频率工作

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset value
CLK_DIV	97h	Clock Divder	-	-	-	-	-	CLKS2	CLKS1	CLKS0	xxxx,x000

CLKS2	CLKS1	CLKS0	分频后CPU的实际工作时钟
0	0	0	系统时钟(外部时钟或内部R/C振荡时钟)
0	0	1	系统时钟/2
0	1	0	系统时钟/4
0	1	1	系统时钟/8
1	0	0	系统时钟/16
1	0	1	系统时钟/32
1	1	0	系统时钟/64
1	1	1	系统时钟/128

1.14 可编程时钟输出

很多实际应用系统需要给外围器件提供时钟,如果单片机能提供可编程时钟输出功能,不但可以降低系统成本,缩小PCB板的面积,而且可以在不需要时钟输出时,关闭时钟输出,这样不但降低了系统的功耗,而且减轻了时钟对外的电磁辐射.STC11/10xx 系列单片机增加了[CLKOUT0/P3.4,CLKOUT1/P3.5,CLKOUT2/P1.0]三个可编程时钟输出脚.CLKOUT0 的输出时钟频率由定时器 0 控制,CLKOUT1 的输出时钟频率由定时器 1 控制,CLKOUT2 的输出时钟频率由独立波特率发生器控制,相应的T0/T1定时器需要工作在定时器的模式2方式(8位自动重装载).

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000
WAKE_CLKO	8Fh	CLK_Output Powerdown_Wakeup Control Register	-	RXD_PIN_IE	T1_PIN_IE	T0_PIN_IE	-	BRTCLKO	T1CLKO	T0CLKO	x000,x000
BRT	9Ch	dedicated Baud-Rate Timer									0000,0000

```
/*    sfr WAKE_CLKO = 0x8F;   // 新增加的特殊功能寄存器
```

如何利用 CLKOUT0/P3.4 和 CLKOUT1/P3.5 管脚输出时钟

CLKOUT0/P3.4 和 CLKOUT1/P3.5 的时钟输出控制由 WAKE_CLKO 寄存器的 T0CLKO 位和 T0CLK1 位控制

CLKOUT0 的输出时钟频率由定时器 0 控制,CLKOUT1 的输出时钟频率由定时器 1 控制, 相应的定时器需要工作在定时器的模式 2 方式(8 位自动重装载模式),不要允许相应的定时器中断,免得 CPU 反复进中断.

新增加的特殊功能寄存器: WAKE_CLKO (地址:0x8F)

```
b7 - NA :
;b6 - RXD_PIN_IE: 1, 允许 RxD/P3.0(或 RxD/P1.6) 下降沿置 RI, 也能使 RxD 脚唤醒 powerdown。
;b5 - T1_PIN_IE : 1, 允许 T1/P3.5 脚下降沿置 T1 中断标志, 也能使 T1 脚唤醒 powerdown。
;b4 - T0_PIN_IE : 1, 允许 T0/P3.4 脚下降沿置 T0 中断标志, 也能使 T0 脚唤醒 powerdown。
;b3 - N/A
b2 - BRTCLKO :1, 允许 P1.0 脚输出时钟, 输出时钟频率 = 1/2 BRT 溢出率
                 BRT 工作在 1T 模式时的输出频率 CLKOUT2  =( Fosc / 2 ) / ( 256 - BRT )
                 BRT 工作在 12T 模式时的输出频率 CLKOUT2 =( Fosc / 2 ) / 12 / ( 256 - BRT )
              0, 不允许 BRT 在 P1.0 脚输出时钟
b1 - T1CLKO  :1, 允许 T1 脚输出 T1(P3.5) 溢出脉冲, 输出时钟频率 = 1/2 T1 溢出率
                 T1 工作在 1T 模式时的输出频率 CLKOUT1 =( Fosc / 2 ) / ( 256 - TH1 )
                 T1 工作在 12T 模式时的输出频率 CLKOUT1 =( Fosc / 2 ) / 12 / ( 256 - TH1 )
              0, 不允许 T1 脚输出 T1(P3.5) 溢出脉冲
b0 - T0CLKO  :1, 允许 T0 脚输出 T0(P3.4) 溢出脉冲, 输出时钟频率 = 1/2 T0 溢出率
                 T0 工作在 1T 模式时的输出频率 CLKOUT0 =( Fosc / 2 ) / ( 256 - TH0 )
                 T0 工作在 12T 模式时的输出频率 CLKOUT0 =( Fosc / 2 ) / 12 / ( 256 - TH0 )
              0, 不允许 T0 脚输出 T0(P3.4) 溢出脉冲
*/
```

如何利用 CLKOUT2/P1.0 管脚输出时钟

CLKOUT2/P1.0 的时钟输出频率:

BRTx12 = 1,独立波特率发生器工作在 1T 模式

CLKOUT2 工作在 1T 模式时的输出频率 CLKOUT2 =(Fosc / 2) / (256 - BRT)

BRTx12 = 0,独立波特率发生器工作在 12T 模式

CLKOUT2 工作在 12T 模式时的输出频率 CLKOUT2 =(Fosc / 2) / 12 / (256 - BRT)

用户在程序中如何具体设置 CLKOUT2/P1.0 管脚输出时钟

1.对 BRT 寄存器独立波特率发生器定时器送 8 位重装载值, BRT = #reload_data

2.对 AUXR 寄存器中的 BRTR 位置 1,让独立波特率发生器定时器运行

3.对 WAKE_CLKO 寄存器中的 BRTCLKO 位置 1,让独立波特率发生器定时器的溢出在 P1.0 口输出时钟

```
/*本程序演示CLKOUT0/INT/T0/P3.4，CLKOUT1/INT/T1/P3.5，CLKOUT2/P1.0输出时钟演示程序*/
/*时钟频率 Fosc = 18.432MHz,T0，T1,独立波特率发生器均工作在12T模式*/
#include"reg51.h"
sfr WAKE_CLKO   = 0x8F;
sfr AUXR        = 0x8E;
sfr  BRT        = 0x9C;
main()
{
/*附加的 SFR WAKE_CLKO (地址：0x8F)
b7 - NA :
;b6 - RXD_PIN_IE: 1，允许 RxD/P3.0(或RxD/P1.6) 下降沿置 RI，也能使 RxD脚唤醒 powerdown。
;b5 - T1_PIN_IE : 1，允许 T1/P3.5 脚下降沿置 T1 中断标志，也能使 T1 脚唤醒 powerdown。
;b4 - T0_PIN_IE : 1，允许 T0/P3.4 脚下降沿置 T0 中断标志，也能使 T0 脚唤醒 powerdown。
b3 - N/A
b2 - BRTCLKO :1，允许P1.0脚输出时钟，输出时钟频率 = 1/2 BRT 溢出率
               BRT 工作在1T模式时的输出频率CLKOUT2  =( Fosc / 2 ) / ( 256 - BRT )
               BRT 工作在12T模式时的输出频率CLKOUT2 =( Fosc / 2 ) / 12 / ( 256 - BRT )
             0，不允许BRT在P1.0脚输出时钟
b1 - T1CLKO  :1，允许 T1 脚输出 T1(P3.5) 溢出脉冲，输出时钟频率 = 1/2 T1 溢出率
               T1工作在1T模式时的输出频率CLKOUT1 =( Fosc / 2 ) / ( 256 - TH1 )
               T1工作在12T模式时的输出频率CLKOUT1 =( Fosc / 2 ) / 12 / ( 256 - TH1 )
             0，不允许 T1 脚输出 T1(P3.5) 溢出脉冲
b0 - T0CLKO  :1，允许 T0 脚输出 T0(P3.4) 溢出脉冲，输出时钟频率 = 1/2 T0 溢出率
               T0工作在1T模式时的输出频率CLKOUT0 =( Fosc / 2 ) / ( 256 - TH0 )
               T0工作在12T模式时的输出频率CLKOUT0 =( Fosc / 2 ) / 12 / ( 256 - TH0 )
             0，不允许 T0 脚输出 T0(P3.4) 溢出脉冲
*/

    TMOD = 0x22;     //T0，T1工作在模式2，8 位自动重装计数器
    AUXR = (AUXR | 0x80);       //T0工作在1T模式
    AUXR = (AUXR | 0x40);      // T1工作在1T模式
    AUXR = (AUXR | 0x04);      // 独立波特率发生器工作在1T模式

    BRT = (256-74);// 对BRT独立波特率发生器定时器送8位重装载值，输出时钟频率124.540KHz
    TH0 = (256-74);// 对T0做时钟输出的8位重装载数，18432000/2/74 = 124540.54约等于125K
    TH1 = (256-240); // 对T1做时钟输出的8位重装载数,输出时钟频率18432000/2/240 = 38400

    WAKE_CLKO = ( WAKE_CLKO | 0x07);  允许T0,T1,独立波特率发生器输出时钟

    TR0 = 1;        // 启动T0开始计数工作，对系统时钟进行分频输出
    TR1 = 1;        // 启动T1开始计数工作，对系统时钟进行分频输出
    AUXR = (AUXR | 0x10);     // 启动独立波特率发生器开始计数工作，对系统时钟进行分频输出
// 至此时钟已经输出，用户可以通过示波器观看到输出的时钟频率
   while(1);
}
```

1.15 新增外部中断,可将CPU从掉电模式唤醒的管脚,远程掉电唤醒

很多实际应用系统中,只有2个外部中断[INT0/P3.2,INT1/P3.3]不够用,STC11/10xx系列单片机可以将[INT/T0/P3.4,INT/T1/P3.5],INT/RxD/P3.0或INT/RxD/P1.6]设置为真正的下降沿中断,这样不但增加了支持外部中断的I/O口数量,而且增加了支持外部掉电唤醒管脚的数量,并且由于[INT/RxD/P3.0或INT/RxD/P1.6]支持下降沿中断,故可实现串口通信远程唤醒.

```
;/* ---  STC International Limited ------------------------------------------------- */
;/* ---  宏晶科技 姚永平  设计 2006/1/6   V1.0 ------------------------------------- */
;/* ---  演示 STC11/10xx 系列 MCU 从掉电模式唤醒 ----------------------------------- */
;如果要在程序中使用或在文章中引用该程序,请在程序中或文章中注明使用了宏晶科技的资料及程序
;*******************************************************************************
;Wake Up Idle and Wake Up Power Down
;*******************************************************************************
;定义 STC11Fxx 系列 MCU 特殊功能寄存器
#include<reg51.h>
;------------------------------------------------------------------------------
;定义特殊功能寄存器
WAKE_CLKO   EQU 8FH
;附加的 SFR WAKE_CLKO (地址：0x8F)
;b7 - NA
;b6 - RXD_PIN_IE: 1，允许 RxD/P3.0(或RxD/P1.6) 下降沿置 RI，也能使 RxD 脚唤醒 powerdown。
;b5 - T1_PIN_IE : 1，允许 T1/P3.5 脚下降沿置 T1 中断标志，也能使 T1 脚唤醒 powerdown。
;b4 - T0_PIN_IE : 1，允许 T0/P3.4 脚下降沿置 T0 中断标志，也能使 T0 脚唤醒 powerdown。
;b3 - N/A

;b2 - BRTCLKO  :1，允许P1.0脚输出时钟，输出时钟频率 = 1/2 BRT 溢出率
                BRT 工作在1T模式时的输出频率CLKOUT2  =( Fosc / 2 ) / ( 256 - BRT )
                BRT 工作在12T模式时的输出频率CLKOUT2 =( Fosc / 2 ) / 12 / ( 256 - BRT )
               0，不允许BRT在P1.0脚输出时钟
;b1 - T1CLKO   :1，允许 T1 脚输出 T1(P3.5) 溢出脉冲，Fck1 = 1/2 T1 溢出率
;b0 - T0CLKO   :1，允许 T0 脚输出 T0(P3.4) 溢出脉冲，Fck0 = 1/2 T1 溢出率
;------------------------------------------------------------------------------
```

所以可将单片机从掉电模式唤醒的管脚如下：

INT0/P3.2，INT1/P3.3，INT/T0/P3.4，INT/T1/P3.5，INT/RxD/P3.0(或INT/RxD/P1.6)

所以以上管脚全部可以作真正的外部中断使用,共有5个真正的外部中断,但有6个外部中断脚,其中RxD/P3.0(或RxD/P1.6)只能够分时复用.

1.16 STC11/10xx系列单片机内部扩展 1024/256字节RAM的使用

1).低128字节的内部RAM（地址：00H～7FH），可直接寻址或间接寻址，(data/idata)

2).高128字节的内部RAM（地址：80H～FFH），只能间接寻址（普通89C51没有），(idata)

3).特殊功能寄存器SFR（地址：80H～FFH），只能直接寻址，(data)

特殊功能寄存器SFR和高128字节的内部RAM是通过寻址方式来区分的，传统的8051系列单片机只有128-256字节RAM供用户使用，在此情况下STC公司响应广大用户的呼声，在一些单片机内部增加了扩展RAM。STC11FxxXE系列单片机内部扩展了1024个字节的扩展RAM，共1280字节RAM。STC10FxxXE系列单片机内部扩展了256个字节的扩展RAM，共512字节RAM。访问内部扩展RAM时，不影响P0口/P2口/P3.6/P3.7/ALE。

STC11Fxx系列单片机8051 单片机 扩展RAM管理及禁止ALE输出 特殊功能寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000

Symbol 符号　Function 功能

XRAM　Internal/External RAM access 内部/外部 RAM 存取

0: 内部扩展的EXT_RAM可以存取.

STC11FxxXE系列单片机

在0000H到03FFH单元,共1024字节,使用MOVX @DPTR指令访问,超过0400H的地址空间总是访问外部数据存储器（含0400H单元），MOVX @Ri只能访问0000H到00FFH单元

STC10FxxXE系列单片机

在0000H到00FFH单元,共256字节,使用MOVX @DPTR指令访问,超过0100H的地址空间总是访问外部数据存储器（含0100H单元），MOVX @Ri只能访问0000H到00FFH单元

1: External data memory access.

外部数据存储器存取,禁止访问内部扩展RAM，此时MOVX @DPTR / MOVX @Ri的使用同普通8052单片机

应用示例供参考（汇编）：

访问内部扩展的 XRAM

```
;新增特殊功能寄存器声明(汇编方式)
AUXR      DATA    8EH;          或者用     AUXR    EQU    8EH    定义
MOV       AUXR,   #00000000B;   XRAM 位清为 ”0 ”,  其实上电复位时此位就为 ”0 ”.
    ;MOVX  A,  @DPTR / MOVX   @DPTR,  A 指令可访问内部扩展的 XRAM
        ;STC10F08XE 系列为(00H - FFH,共 256 字节)
    ;MOVX  A,  @Ri / MOVX  A, @Ri 指令可直接访问内部扩展的 XRAM
        ;使用此指令 STC10F08XE 系列 只能访问内部扩展的 XRAM(00H - FFH,共 256 字节)
;写芯片内部扩展的 XRAM
    MOV       DPTR,   #address
    MOV       A,  #value
    MOVX      @DPTR,  A
;读芯片内部扩展的 XRAM
    MOV       DPTR,   #address
    MOVX      A,  @DPTR
STC11F32XE 系列
; 如果 #address < 3FFH，则在 XRAM 位为 ”0 ”时，访问物理上在内部，逻辑上在外部的此 XRAM
; 如果 #address >= 400H，则总是访问物理上外部扩展的 RAM 或 I/O 空间（400H--FFFFH）
```

禁止访问内部扩展的 XRAM ,以防冲突

```
MOV       AUXR,   #00000010B;   XRAM 控制位设置为 ”1 ”,  禁止访问 XRAM,以防冲突
```

有些用户系统因为外部扩展了 I/O 或者用片选去选多个 RAM 区,有时与此内部扩展的 XRAM 逻辑地址上有冲突，将此位设置为 ”1 ”, 禁止访问此内部扩展的 XRAM 就可以了.

大实话 ： 其实不用设置 AUXR 寄存器即可直接用 MOVX @DPTR 指令访问此内部扩展的 XRAM,超过此 RAM 空间,将访问片外单元.如果系统外扩了 SRAM,而实际使用的空间小于 1024 字节,则可直接将此 SRAM 省去,比如省去 STC62WV256,IS62C256,UT6264 等.

应用示例供参考（C 语言）：

```
/*  访问内部扩展的 XRAM */
/*  STC11F32XE 系列单片机为(00H - 3FFH，共 1024 字节扩展的 XRAM) */
    /*  新增特殊功能寄存器声明(C 语言方式)    */
    sfr AUXR=   0x8e/*  如果不需设置 AUXR 就不用声明 AUXR       */
    AUXR    =   0x00;   /*  0000,0000  XRAM 位清 0，其实上电复位时此位就为 0 */
    unsigned char  xdata  sum,   loop_counter, test_array[128];
    /*  将变量声明成 xdata 即可直接访问此内部扩展的 XRAM    */
```

```
/*  写芯片内部扩展的 XRAM    */
    sum =   0;
    loop_counter =   128;
    test_array[0]    =   5;
/*  读芯片内部扩展的 XRAM    */
    sum =   test_array[0];
/*  STC11F32XE 系列:
        如果 #address <3FFH，则在 XRAM 位为 ”0 ” 时，访问物理上在内部，逻辑
    上在外部的此 XRAM
        如果 #address>=400H，则总是访问物理上外部扩展的 RAM 或 I/O 空间（400H-FFFFH）
*/
```

禁止访问内部扩展的 XRAM，以防冲突

```
AUXR=   0x02;   /*  0000,0010，XRAM 位设为 ”1 ”，禁止访问 XRAM,以防冲突 */
```

有些用户系统因为外部扩展了 I/O 或者用片选去选多个 RAM 区,有时与此内部扩展的 XRAM 逻辑上有冲突，将此位设置为 ”1 ”， 禁止访问此内部扩展的 XRAM 就可以了.

STC11/10xx 系列单片机内部扩展 RAM 演示程序

```
;/* ---  STC International Limited ------------------------------------------ */
;/* ---  演示 STC11/10xx 系列单片机 MCU 内部扩展 RAM 演示程序 ------------------- */
;/* ---  本演示程序在 STC-ISP Ver 3.0A.PCB 的下载编程工具上测试通过 ----------- */



#include <reg51.h>
#include <intrins.h>        /* use _nop_() function */

sfr AUXR = 0x8e;

sbit ERROR_LED = P1^5;
sbit OK_LED = P1^7;
void main()
{
    unsigned int array_point = 0;

    /* 测试数组 Test_array_one[128],Test_array_two[128]*/
    unsigned char xdata Test_array_one[128]        =
    {
        0x00,     0x01,     0x02,     0x03,     0x04,     0x05,     0x06,     0x07,
        0x08,     0x09,     0x0a,     0x0b,     0x0c,     0x0d,     0x0e,     0x0f,
        0x10,     0x11,     0x12,     0x13,     0x14,     0x15,     0x16,     0x17,
        0x18,     0x19,     0x1a,     0x1b,     0x1c,     0x1d,     0x1e,     0x1f,
        0x20,     0x21,     0x22,     0x23,     0x24,     0x25,     0x26,     0x27,
         0x28,     0x29,     0x2a,     0x2b,     0x2c,     0x2d,     0x2e,     0x2f,
        0x30,     0x31,     0x32,     0x33,     0x34,     0x35,     0x36,     0x37,
        0x38,     0x39,     0x3a,     0x3b,     0x3c,     0x3d,     0x3e,     0x3f,
        0x40,     0x41,     0x42,     0x43,     0x44,     0x45,     0x46,     0x47,
        0x48,     0x49,     0x4a,     0x4b,     0x4c,     0x4d,     0x4e,     0x4f,
        0x50,     0x51,     0x52,     0x53,     0x54,     0x55,     0x56,     0x57,
        0x58,     0x59,     0x5a,     0x5b,     0x5c,     0x5d,     0x5e,     0x5f,
        0x60,     0x61,     0x62,     0x63,     0x64,     0x65,     0x66,     0x67,
        0x68,     0x69,     0x6a,     0x6b,     0x6c,     0x6d,     0x6e,     0x6f,
        0x70,     0x71,     0x72,     0x73,     0x74,     0x75,     0x76,     0x77,
        0x78,     0x79,     0x7a,     0x7b,     0x7c,     0x7d,     0x7e,     0x7f
     }
```

```
    unsigned char xdata Test_array_two[128]        =
    {
        0x00,     0x01,     0x02,     0x03,     0x04,     0x05,     0x06,     0x07,
        0x08,     0x09,     0x0a,     0x0b,     0x0c,     0x0d,     0x0e,     0x0f,
        0x10,     0x11,     0x12,     0x13,     0x14,     0x15,     0x16,     0x17,
        0x18,     0x19,     0x1a,     0x1b,     0x1c,     0x1d,     0x1e,     0x1f,
        0x20,     0x21,     0x22,     0x23,     0x24,     0x25,     0x26,     0x27,
        0x28,     0x29,     0x2a,     0x2b,     0x2c,     0x2d,     0x2e,     0x2f,
        0x30,     0x31,     0x32,     0x33,     0x34,     0x35,     0x36,     0x37,
        0x38,     0x39,     0x3a,     0x3b,     0x3c,     0x3d,     0x3e,     0x3f,
        0x40,     0x41,     0x42,     0x43,     0x44,     0x45,     0x46,     0x47,
        0x48,     0x49,     0x4a,     0x4b,     0x4c,     0x4d,     0x4e,     0x4f,
        0x50,     0x51,     0x52,     0x53,     0x54,     0x55,     0x56,     0x57,
        0x58,     0x59,     0x5a,     0x5b,     0x5c,     0x5d,     0x5e,     0x5f,
        0x60,     0x61,     0x62,     0x63,     0x64,     0x65,     0x66,     0x67,
        0x68,     0x69,     0x6a,     0x6b,     0x6c,     0x6d,     0x6e,     0x6f,
        0x70,     0x71,     0x72,     0x73,     0x74,     0x75,     0x76,     0x77,
        0x78,     0x79,     0x7a,     0x7b,     0x7c,     0x7d,     0x7e,     0x7f,
    };
    ERROR_LED = 1;
    OK_LED = 1;
    for(array_point=0; array_point<512; array_point++)
    {
        if(Test_array_one[array_point]!=Test_array_two [array_point]){
            ERROR_LED = 0;
            OK_LED = 1;
            break;
        }
        else{
             OK_LED = 0;
             ERROR_LED = 1;
             }
     }
    while(1);
}
```

1.17 双数据指针 DPTR0,DPTR1 的使用

STC11/10xx 系列 8051 单片机 双数据指针 特殊功能寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR1	A2h	Auxiliary Register 1	UART_P1	-	-	-	GF2	-	-	DPS	0xxx,0xx0

Symbol 符号 Function 功能

DPS DPTR registers select bit. DPTR 寄存器选择位

0: DPTR0 is selected DPTR0 被选择

1: DPTR1 is selected DPTR1 被选择

此系列单片机有两个 16-bit 数据指针,DPTR0,DPTR1.当 DPS 选择位为 0 时,选择 DPTR0, 当 DPS 选择位为 1 时,选择 DPTR1.

AUXR1 特殊功能寄存器,位于 A2H 单元,其中的位不可用布尔指令快速访问.但由于 DPS 位位于 bit0,故对 AUXR1 寄存器用 INC 指令,DPS 位便会反转,由 0 变成 1 或由 1 变成 0,即可实现双数据指针的快速切换.

应用示例供参考:

```
;新增特殊功能寄存器定义
AUXR1   DATA    0A2H
MOV     AUXR1,  #0      ;此时 DPS 为 0,DPTR0 有效

MOV     DPTR,   #1FFH   ;置 DPTR0 为 1FFH
MOV     A,      #55H
MOVX    @DPTR,  A       ;将 1FFH 单元置为 55H

MOV     DPTR,   #2FFH   ; 置 DPTR0 为 2FFH
MOV     A,      #0AAH
MOVX    @DPTR,  A       ;将 2FFH 单元置为 0AAH

INC     AUXR1           ; 此时 DPS 为 1,DPTR1 有效
MOV     DPTR,   #1FFH   ; 置 DPTR1 为 1FFH
MOVX    A,      @DPTR   ; 读 DPTR1 数据指针指向的 1FFH 单元的内容,累加器 A 变为 55H.

INC     AUXR1           ; 此时 DPS 为 0,DPTR0 有效
MOVX    A,      @DPTR   ; 读 DPTR0 数据指针指向的 2FFH 单元的内容,累加器 A 变为 0AAH.

INC     AUXR1           ; 此时 DPS 为 1,DPTR1 有效
MOVX    A,      @DPTR   ; 读 DPTR1 数据指针指向的 1FFH 单元的内容,累加器 A 变为 55H.

INC     AUXR1           ; 此时 DPS 为 0,DPTR0 有效
MOVX    A,      @DPTR   ; 读 DPTR0 数据指针指向的 2FFH 单元的内容,累加器 A 变为
```

1.18 STC11Fxx系列单片机片外64K数据总线速度控制

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
BUS_SPEED	A1h	Bus-Speed Control	-	-	ALES1	ALES0	-	RWS2	RWS1	RWS0	xx10,x011

ALES1	ALES0	
0	0	P0地址建立时间和保持时间到ALE信号的下降沿是1个时钟
0	1	P0地址建立时间和保持时间到ALE信号的下降沿是2个时钟
1	0	P0地址建立时间和保持时间到ALE信号的下降沿是3个时钟(复位之后默认设置)
1	1	P0地址建立时间和保持时间到ALE信号的下降沿是4个时钟

RWS2	RWS1	RWS0	
0	0	0	MOVX 读/写 脉冲是1个时钟
0	0	1	MOVX 读/写 脉冲是2个时钟
0	1	0	MOVX 读/写 脉冲是3个时钟
0	1	1	MOVX 读/写 脉冲是4个时钟(复位之后默认设置)
1	0	0	MOVX 读/写 脉冲是5个时钟
1	0	1	MOVX 读/写 脉冲是6个时钟
1	1	0	MOVX 读/写 脉冲是7个时钟
1	1	1	MOVX 读/写 脉冲是8个时钟

当MOVX 指令访问物理上在内部，逻辑上在外部的片内扩展的256字节EXT_RAM时，以上设置均被忽略，以上设置只是在访问真正的片外扩展器件时有效。

助记符	功能说明	字节数	1时钟/机器周期 单片机所需时钟	效率提升
MOVX A,@Ri	逻辑上在外部的片内扩展RAM，（8位地址）送入累加器	1	4	6倍
MOVX A,@DPTR	逻辑上在外部的片内扩展RAM，（16位地址）送入累加器	1	3	8倍
MOVX @Ri,A	累加器送逻辑上在外部的片内扩展RAM（8位地址）	1	3	8倍
MOVX @DPTR ,A	累加器送逻辑上在外部的片内扩展RAM（16位地址）	1	3	8倍
MOVX A,@Ri	物理上在外部的片外扩展RAM，（8位地址）送入累加器	1	7 + ?	*Note1
MOVX A,@DPTR	物理上在外部的片外扩展RAM，（16位地址）送入累加器	1	7 + ?	*Note1
MOVX @Ri,A	累加器送物理上在外部的片外扩展RAM，（8位地址）	1	7 + ?	*Note1
MOVX @DPTR ,A	累加器送物理上在外部的片外扩展RAM，（16位地址）	1	7 + ?	*Note1

Note1:访问物理上在片外的扩展RAM所需时钟：7 + 2 x ALE_Bus_Speed + RW_Bus_Speed

其中ALE_Bus_Speed由BUS_SPEED控制寄存器中的ALES1/ALES0决定(建议使用缺省设置,也可以设置更快的总线访问速度,但不要设置BUS_SPEED = 00H)

其中RW_Bus_Speed由BUS_SPEED控制寄存器中的RWS2/RWS1/RWS0决定

1.19 STC11/10xx 系列单片机P4口的使用

STC11/10xx 系列单片机与 P4 口有关的特殊功能寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
P4	C0h	8 - bit Port 4	P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0	1111,1111
P4M1	B3h	P4 Configuration 1									0000,0000
P4M0	B4h	P4 Configuration 0									0000,0000
P4SW	BBh	Port - 4 switch	-	NA_P4.6	ALE_P4.5	NA_P4.4	-	-	-	-	x000,xxxx

对 STC11/10xx 系列单片机的 P4 口的访问，如同访问常规的 P1/P2/P3 口，并且均可位寻址，P4 的地址

C0H P4 端口的地址在C0h，P4口中的每一位均可位寻址，位地址如下：

位	P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0
位地址	C7h	C6h	C5h	C4h	C3h	C2h	C1h	C0h

由 P4SW 寄存器设置(NA/P4.4，ALE/P4.5，NA/P4.6)三个端口的第二功能

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
P4SW	BBh	Port - 4 switch	-	NA_P4.6	ALE_P4.5	NA_P4.4	-	-	-	-	x000,xxxx

NA/P4.4: 0,复位后 P4SW.4 = 0,NA/P4.4 脚是弱上拉，无任何功能
1,通过设置P4SW.4 = 1,将NA/P4.4脚设置成I/O口(P4.4)

ALE/P4.5: 0,复位后P4SW.5 = 0,ALE/P4.5脚是ALE信号,只有在用MOVX指令访问片外扩展器件时才有信号输出
1，通过设置 P4SW.5 = 1,将 ALE/P4.5 脚设置成 I/O 口(P4.5)

NA/P4.6: 0,复位后 P4SW.4 = 0,NA/P4.4 脚是弱上拉，无任何功能
1,通过设置P4SW.6 = 1将NA/P4.6脚设置成I/O口(P4.6)

在 ISP 烧录程序时设置 LQFP44/PDIP40 封装的单片机 RST/P4.7 管脚的第二功能，

RST/P4.7 在 ISP 烧录程序时选择是复位脚还是 P4.7 口，如设置成 P4.7 口，必须使用外部时钟。

新增 P4 口特殊功能寄存器如何声明地址，举例如下：

汇编语言(新增 P4 口地址声明) ：

```
P4    EQU  0C0H
P40   EQU  0C0H
P41   EQU  0C1H
P47   EQU  0C7H
```

C 语言 (新增 P4 口地址声明) ：

```
sfr  P4  = 0xC0;     //P4口寄存器的字节地址
sbit  P40 = 0xC0;    //P4.0口的位地址
sbit  P41 = 0xC1;    //P4.1口的位地址
sbit  P42 = 0xC2;   //P4.2口的位地址
sbit  P47 = 0xC7;   //P4.7口的位地址
```

1.20 STC11/10xx 系列单片机串行口在 P3 口还是在 P1 口的使用

STC11/10xx 系列 8051 单片机 串行口选择控制 特殊功能寄存器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR1	A2h		UART_P1	-	-	-	GF2	-	-	DPS	0xxx,0xx0

UART_P1: 0， 串口 /UART 在 P3 口[RxD/P3.0,TxD/P3.1]
1， 串口 /UART 在 P1 口，将串口从 P3 口切换到 P1 口[RxD/P1.6,TxD/P1.7]

串行口做主机通信时，可控制串口通信在[RxD/P3.0,TxD/P3.1]和[RxD/P1.6 口,TxD/P1.7]之间任意切换，实现 2 组串口。建议用户将自己的串行口设置在[RxD/P1.6 口,TxD/P1.7]而将[RxD/P3.0,TxD/P3.1]口作为 ISP 下载的专用通信口,当然也可以当用户的普通 I/O 口用

1.21 串行口1使用独立波特率发生器作为波特率发生器

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
BRT	9Ch	dedicated Baud-Rate timer 独立波特率发生器定时器,装入重装载数									0000,0000
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000
SCON	98h	Serial Control	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
SBUF	99h	Serial Data Buffer									xxxx,xxxx
PCON	87h	Power Control	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
IE	A8h	Interrupt Enable	EA	ELVD	-	ES	ET1	EX1	ET0	EX0	00x0,0000
IP	B8h	Interrupt Priority Low	-	PLVD	-	PS	PT1	PX1	PT0	PX0	x0x0,0000
IPH	B7h	Interrupt Priority High	-	PLVDH	-	PSH	PT1H	PX1H	PT0H	PX0H	x0x0,0000
SADEN	B9h	Slave Address Mask									0000,0000
SADDR	A9h	Slave Address									0000,0000
以下是使用定时器1作为串口1的波特率发生器时需要用到的寄存器，现在可以不用了											
TCON	88h	Timer / Counter 0 and 1 Control	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
TMOD	89h	Timer / Counter 0 and 1 Modes	GATE GATE1	C/T# C/T1#	M1 M1_1	M0 M1_0	GATE GATE0	C/T# C/T0#	M1 M0_1	M0 M0_0	0000,0000
TH1	8Dh	Timer / Counter 1 High Byte									0000,0000
TL1	8Bh	Timer / Counter 1 Low Byte									0000,0000

当设置 AUXR 寄存器中的 S1BRS 位(串口 1 波特率选择位)为 1 时，串口 1 选择独立波特率发生器作为波特率发生器,此时定时器 1 可以释放出来作为定时器 / 计数器 / 时钟输出使用.

SM0	SM1	方式	功能说明	串口1波特率
0	0	0	同步移位串行方式	Fosc / 12，UART_M0x6 = 1时，波特率是Fosc / 2
0	1	1	8位UART，波特率可变	(2^{SMOD} / 32) x BRT独立波特率发生器的溢出率
1	0	2	9位UART	(2^{SMOD} / 64) x Fosc系统工作时钟频率
1	1	3	9位UART，波特率可变	(2^{SMOD} / 32) x BRT独立波特率发生器的溢出率
BRT独立波特率发生器的溢出率 = Fosc/12/(256 - BRT),当BRTx12 = 0时 BRT独立波特率发生器的溢出率 = Fosc / (256 - BRT),当BRTx12 = 1时				

串行口做主机通信时，可控制串口通信在[RxD/P3.0,TxD/P3.1]和[RxD/P1.6,TxD/P1.7.]之间任意切换，实现 2 组串口。建议用户将自己的串行口设置在[RxD/P1.6,TxD/P1.7.]而将[RxD/P3.0,TxD/P3.1]口作为 ISP 下载的专用通信口,当然也可以当用户的普通 I/O 口用

串口 1 模式 0 ：

串行数据通过 RxD/P3.0 接收,TxD/P3.1 输出同步移位时钟，发送接收的是八位数据,低位在先,波特率固定在 Fosc / 12，忽略波特率发生器

串口 1 波特率在模式 0 = Fosc 系统工作时钟频率 / 12

串口 1 模式 1：

10 位数据通过 TxD/P3.1 发送，通过 RxD/P3.0 接收。一帧数据包含一个起始位(0)，8 个数据位(低位在先)，和一个停止位(1)。接收时，停止位进入特殊功能寄存器 SCON 的 RB8 位。波特率由独立波特率发生器 BRT 的溢出率决定。

串口 1 波特率在模式 1 = (2^{SMOD} / 32) x BRT 独立波特率发生器的溢出率

当 SMOD = 0 时，串口 1 波特率 = BRT 独立波特率发生器的溢出率 / 32，

当 SMOD = 1 时，串口 1 波特率 = BRT 独立波特率发生器的溢出率 / 16，

BRT 独立波特率发生器的溢出率 = Fosc/12/(256 - BRT),当 BRTx12 = 0 时，

BRT 独立波特率发生器的溢出率 = Fosc / (256 - BRT),当 BRTx12 = 1 时

串口 1 模式 2：

11 位数据通过 TxD/P3.1 发送，通过 RxD/P3.0 接收。一帧数据包含一个起始位(0)，8 个数据位(低位在先)，一个可编程的第 9 位，和一个停止位(1)。发送时，第 9 位数据位来自特殊功能寄存器 SCON 的 TB8 位.接收时，第 9 位进入特殊功能寄存器 SCON 的 RB8 位。波特率可编程为系统时钟频率：Fosc / 32 或者 Fosc / 64

串口 1 波特率在模式 2 = (2^{SMOD} / 64) x Fosc 系统工作时钟频率

当 SMOD = 0 时，串口 1 波特率 = Fosc 系统工作时钟频率 / 64

当 SMOD = 1 时，串口 1 波特率 = Fosc 系统工作时钟频率 / 32

串口 1 模式 3：

波特率是可变的，其它和模式 2 相同

11 位数据通过 TxD/P3.1 发送，通过 RxD/P3.0 接收。一帧数据包含一个起始位(0)，8 个数据位(低位在先)，一个可编程的第 9 位，和一个停止位(1)。发送时，第 9 位数据位来自特殊功能寄存器 SCON 的 TB8 位.接收时，第 9 位进入特殊功能寄存器 SCON 的 RB8 位。

串口 1 波特率在模式 3 = (2^{SMOD} / 32) x BRT 独立波特率发生器的溢出率

当 SMOD = 0 时，串口 1 波特率 = BRT 独立波特率发生器的溢出率 / 32，

当 SMOD = 1 时，串口 1 波特率 = BRT 独立波特率发生器的溢出率 / 16，

BRT 独立波特率发生器的溢出率 = Fosc/12/(256 - BRT),当 BRTx12 = 0 时，

BRT 独立波特率发生器的溢出率 = Fosc / (256 - BRT),当 BRTx12 = 1 时

用户在程序中如何具体使用串口 1 和独立波特率发生器 BRT

1.设置串口 1 的工作模式，SCON 寄存器中的 SM0 和 SM1 两位决定了串口 1 的 4 种工作模式。

2.设置串口 1 的波特率,使用独立波特率发生器寄存器和相应的位：

BRT 独立波特率发生器寄存器,BRTx12 位，SMOD 位

3.启动独立波特率发生器，让 BRTR 位为 1，BRT 独立波特率发生器寄存器就立即开始计数。

4.设置串口 1 的中断优先级，及打开中断相应的控制位是:

PS,PSH,ES,EA

5.如要串口 1 接收，将 REN 置 1 即可

如要串口 1 发送，将数据送入 SBUF 即可，

接收完成标志 RI,发送完成标志 TI,要由软件清 0。

```
;当串口工作在模式 1 和模式 3 时,计算相应的波特率需要设置的重装载数,结果送入 BRT 寄存器
;计算自动重装数 RELOAD (SMOD = 0, SMOD 是 PCON 特殊功能寄存器的最高位):
;  1. 计算 RELOAD (以下是 SMOD = 0 时的计算公式)
;
;     a) 12T 模式的计算公式： RELOAD = 256 - INT(Fosc/Baud0/32/12 + 0.5)
;     b)  1T 模式的计算公式： RELOAD = 256 - INT(Fosc/Baud0/32 + 0.5)
计算出的 RELOAD 数直接送 BRT 寄存器
;
;     式中: INT() 表示取整运算即舍去小数, 在式中加 0.5 可以达到四舍五入的目的
;           Fosc = 晶振频率
;           Baud0 = 标准波特率
;
;  2. 计算用 RELOAD 产生的波特率:
;       a)   Baud = Fosc/(256 - RELOAD)/32/12       12T 模式
;       b)   Baud = Fosc/(256 - RELOAD)/32           1T 模式
;
;  3. 计算误差
;          error = (Baud - Baud0)/Baud0 * 100%
;  4. 如果误差绝对值 > 3% 要更换波特率或者更换晶体频率, 重复步骤 1-4
;
;
;例: Fosc = 22.1184MHz, Baud0 = 57600 (12T 模式)
; 1. RELOAD = 256 - INT( 22118400/57600/32/12 + 0.5)
;           = 256 - INT( 1.5 )
;           = 256 - 1
;           = 255
;           = 0FFH
; 2. Baud = 22118400/(256-255)/32/12
;         = 57600
; 3. 误差等于零
```

```
;例: Fosc = 18.432MHz, Baud0 = 57600 (12T 模式)
; 1. RELOAD = 256 - INT( 18432000/57600/32/12 + 0.5)
;           = 256 - INT( 0.833 + 0.5 )
;           = 256 - INT( 1.333 )
;           = 256 - 1
;           = 255
;           = 0FFH
; 2. Baud = 18432000/(256-255)/32/12
;         = 48000
; 3. error = (48000 - 57600)/57600 * 100%
;          = -16.66%
; 4. 误差很大, 要更换波特率或者更换晶体频率, 重新计算请见下一例

;例: Fosc = 18.432MHz, Baud0 = 9600 (12T 模式)
; 1. RELOAD = 256 - INT( 18432000/9600/32/12 + 0.5)
;           = 256 - INT( 5.5 )
;           = 256 - 5
;           = 251
;           = 0FBH
; 2. Baud = 18432000/(256-251)/32/12
;         = 9600
; 3. 一目了然, 误差等于零

;例: Fosc = 2.000MHz, Baud = 4800 (1T 模式)
; 1. RELOAD = 256 - INT( 2000000/4800/32 + 0.5)
;           = 256 - INT( 13.02 + 0.5 )
;           = 256 - INT( 13.52 )
;           = 256 - 13
;           = 243
;           = 0F3H
; 2. Baud = 2000000/(256-243)/32
;         = 4808
; 3. error = 0.16%
;------------------------------------------------------------------------------
```

1.22 使用独立波特率发生器作串行通信测试程序

```
;/* ---  STC International Limited ------------------------------------------------- */
;/* ---  宏晶科技 姚永平  设计 2007/1/6   V1.0 ------------------------------------- */
;/* ---  演示 STC11/10xx 系列 MCU 看门狗及其溢出时间计算公式 ---------------------- */
#include<reg51.h>
#include<intrins.h>
sfr  AUXR      =    0x8e;
sfr  AUXR1     =    0xA2;
sfr  BRT       =    0x9c;
sbit MCU_Start_Led  =  P1^4;
//unsigned char array[9] = {0,2,4,6,8,10,12,14,16};
unsigned char array[9] = {0x00,0x02,0x04,0x06,0x08,0x0A,0x0C,0x0E,0x10};
#define   RELOAD_COUNT   0xfb   //18.432MHz,12T,SMOD=0,9600bps

void serial_port_initial();
void send_UART(unsigned char);
void UART_Interrupt_Receive(void);
void delay(void);
void display_MCU_Start_Led(void);

void main(void)
{
     unsigned  char i = 0;
     serial_port_initial();       //串口初始化
     display_MCU_Start_Led();     //点亮发光二极管表示单片机开始工作
     send_UART(0x34);             //串口发送数据表示单片机串口正常工作
     send_UART(0xa7);             //串口发送数据表示单片机串口正常工作
     for(i = 0;i<9;i++)
     {
          send_UART(array[i]);
     }
     while(1);
}
```

```
/*
void serial_port_initial()     //使用定时器1作为波特率发生器
{
    SCON     =     0x50;          //0101,0000 8位可变波特率，无奇偶校验位
    TMOD     =     0x21;          //0011,0001 设置定时器1为8位自动重装计数器
    TH1      =     RELOAD_COUNT; //设置定时器1自动重装数
    TL1      =     RELOAD_COUNT;
    TR1      =     1;     //开定时器1
    ES       =     1;     //允许串口中断
    EA       =     1;     //开总中断
}
*/
void serial_port_initial()       //使用独立波特率发生器作为波特率发生器
{
    SCON     =    0x50;    //0101,0000 8位可变波特率，无奇偶校验位
    BRT      =    RELOAD_COUNT;
    AUXR     =    0x11;
                  // T0x12,T1x12,UART_M0x6,BRTR,S2SMOD,BRTx12,XRAM,S1BRS
                  // Baud = Fosc/(256 - RELOAD_COUNT)/32/12 (12T模式)
                  // Baud = Fosc/(256 - RELOAD_COUNT)/32    (1T模式)
                  // BRTR = 1,启动独立波特率发生器
                  // S1BRS = 1,串口1选择独立波特率发生器作为波特率发生器，
                 //  此时定时器1可以释放出来作为定时器，计数器，时钟输出使用
//   AUXR1    =    0x80; //释放该行指令，则串行口从P3口切换到P1口
    ES       =    1;     //允许串口中断
    EA       =    1;     //开总中断
}

void send_UART(unsigned char i)
{
    ES       =    0;  //关串口中断
    TI       =    0;  //清零串口发送完成中断请求标志
    SBUF     =    i;
    while(TI ==0);    //等待发送完成
    TI       =    0;  //清零串口发送完成中断请求标志
    ES       =    1;  //允许串口中断
}
```

```
void UART_Interrupt_Receive(void) interrupt 4
{
     unsigned char   k   =   0;
     if(RI==1)
    {
         RI  =   0;
         k   =   SBUF;
         send_UART(k+1);
    }
    else
    {
         TI  =  0;
    }
}
void delay(void)
{
     unsigned int j  =   0;
     unsigned int g  =   0;
     for(j=0;j<5;j++)
     {
         for(g=0;g<50000;g++)
         {
           _nop_();
           _nop_();
           _nop_();
         }
     }
}
void display_MCU_Start_Led(void)
{
    unsigned  char i = 0;
    for(i=0;i<5;i++)
    {
        MCU_Start_Led   =   0;  //顶亮MCU开始工作指示灯
        delay();
        MCU_Start_Led   =   1;  //熄灭MCU开始工作指示灯
        delay();
        MCU_Start_Led   =   0;  //顶亮MCU开始工作指示灯
    }
}
```

1.23 每个单片机具有全球唯一身份证号码(ID号)

宏晶科技最新一代STC11/10xx系列每一个单片机出厂时都具有全球唯一身份证号码(ID号)，用户可以在单片机上电后读取内部RAM单元从F1H - F7H连续7个单元的值来获取此单片机的唯一身份证号码(ID号),使用“ MOV @Ri ”指令来读取。

1.24 如何知道单片机内部R/C振荡器频率(内部时钟频率)

宏晶科技最新一代STC11/10xx系列单片机除了可以使用传统的外部时钟外，还可以选择内部R/C振荡器时钟源(内部时钟).如果选择单片机工作在内部R/C振荡器频率(内部时钟频率),则可以省掉外部晶振。这时XTAL1/XTAL2浮空.但由于使用内部时钟源误差较大，所以在对时序要求较高或者有串行通信的情况下不建议使用内部R/C时钟源。在上电初始化程序时,我们可以通过读取内部RAM单元（FCH,FDH,FEH,FFH连续四个单元)的值来获取单片机出厂时的内部R/C振荡器频率(内部时钟频率)。可以通过读取内部RAM单元(F8H,F9H,FAH,FBH连续四个单元)的值来获取用户最后一次使用内部R/C振荡器时钟下载程序时的频率(内部时钟频率),使用“ MOV @Ri ”指令来读取。

1.25: STC11/10xx系列单片机取代传统8051单片机注意事项

STC11/10xx 系列单片机的定时器 0/ 定时器 1/ 串行口与传统 8051 完全兼容，上电复位后，定时器部分缺省还是除 12 再计数的，而串口由定时器 1 控制速度，所以，定时器 / 串口完全兼容。

增加了独立波特率发生器，省去了传统 8052 的定时器 2，如是用 T2 做波特率的，请改用独立波特率发生器做波特率发生器。

传统 8051 的 111 条指令执行速度全面提速，最快的指令快 24 倍，最慢的指令快 3 倍.靠软件延时实现精确延时的程序需要调整。

其它需注意的细节：

ALE:

传统 8051 单片机的 ALE 脚对系统时钟进行 6 分频输出，可对外提供时钟，STC11/10xx 系列不对外输出时钟，如果传统设计利用 ALE 脚对外输出时钟，请利用 STC11/10xx 系列的可编程时钟输出脚对外输出时钟(CLKOUT0/CLKOUT1/CLKOUT2)或 XTAL2 脚串一个 200 欧姆电阻对外输出时钟.

传统 8051 单片机时钟频率较高时，ALE 脚是一个干扰源，所以 STC89 系列单片机增加了 AUXR 特殊功能寄存器，其中的 Bit0/ALEOFF 位允许禁止 ALE 对系统时钟分频输出。而 STC11/10xx 单片机直接禁止 ALE 脚对系统时钟进行 6 分频输出，彻底清除此干扰源.也有利于系统的抗干扰设计.请自行比较如下的寄存器.

STC89 系列的 AUXR 寄存器:

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset value
AUXR	8Eh	Auxiliary Register 0	-	-	-	-	-	-	EXTRAM	ALEOFF	xxxx,xx00

ALEOFF 0：ALE 脚对系统时钟进行 6 分频输出

1：ALE 脚仅在对外部 64K 数据总线进行 MOVX 指令时才有地址锁存信号输出

STC11/10xx 系列的 AUXR 寄存器:

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000

S1BRS: 0,缺省，串口 1 波特率发生器选择定时器 1，S1BRS 是串口 1 波特率发生器选择位

1,独立波特率发生器作为串口 1 的波特率发生器，此时定时器 1 与串口无关

PSEN:

传统 8031/8032 有 PSEN 信号可以跑外部程序，可以外扩外部程序存储器.现在 STC11/10xx 系列单片机由于是系统晶片概念，内部有大容量程序存储器，不需外扩外部程序存储器，所以直接将PSEN信号去除，可以当普通I/O 口使用.

普通 I/O 口既作为输入又作为输出:

传统8051单片机执行I/O口操作，由高变低或由低变高，以及读外部状态都是12个时钟，而现在STC11/10xx系列单片机执行相应的操作是 4 个时钟.传统 8051 单片机如果对外输出为低，直接读外部状态是读不对的.必须先将 I/O 口置高才能够读对，而传统 8051 单片机由低变高的指令是 12 个时钟，该指令执行完成后，该 I/O 口也确实已变高.故可以紧跟着由低变高的指令后面，直接执行读该 I/O 口状态指令.而 STC11/10xx 系列单片机由于执行由低变高的指令是4个时钟，太快了，相应的指令执行完以后，I/O口还没有变高，要再过一个时钟之后，该 I/O 口才可以变高.故建议此状况下增加 2 个空操作延时指令再读外部口的状态.

P4 口:

最新 STC11/10xx 系列单片机 P4 口地址在 C0H，有完整的 P4 口(P4.0-P4.7)，未扩展外部 INT2/INT3 中断

传统 STC89 系列单片机的 P4 口地址在 E8H，P4 口只有一半(P4.0-P4.3)，P4 有扩展外部 INT2/INT3 中断

如需要 STC11/10 系列单片机的高速性能，又需要在 P4 口上增加 2 个外部中断，请使用 STC12C5Axx 系列单片机

I/O 口驱动能力:

最新 STC11/10xx 系列单片机 I/O 口的灌电流是 20mA，驱动能力超强，驱动大电流时，不容易烧坏.

传统 STC89Cxx 系列单片机 I/O 口的灌电流是 6mA，驱动能力不够强，不能驱动大电流，建议使用 STC11/10xx 系列

中断优先级：

最新 STC11/10xx 系列单片机中断优先级是 2 级，兼容传统 8051

传统 STC89 系列增强型单片机中断优先级是 4 级，增加了 IPH 寄存器，与 IPH 寄存器组合使用，支持 4 级优先级

如需要 STC11/10 系列单片机的高速性能，又需要 4 级中断优先级，请使用 STC12C5Axx 系列单片机

看门狗：

最新 STC11/10xx 系列单片机的看门狗寄存器 WDT_CONTR 的地址在 C1H，增加了看门狗复位标志位

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WDT_CONTR	C1h	Watch-Dog-Timer Control register	WDT_FLAG	-	EN_WDT	CLR_WDT	IDLE_WDT	PS2	PS1	PS0	xx00,0000

传统 STC89 系列增强型单片机看门狗寄存器 WDT_CONTR 的地址在 E1H，没有看门狗复位标志位

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WDT_CONTR	E1h	Watch-Dog-Timer Control register	-	-	EN_WDT	CLR_WDT	IDLE_WDT	PS2	PS1	PS0	xx00,0000

最新STC11/10xx系列单片机的看门狗在ISP烧录程序可设置上电复位后直接启动看门狗，而传统STC89系列单片机无此功能.故最新 STC11/10xx 系列单片机看门狗更可靠.

EEPROM

STC11/10xx单片机ISP/IAP控制寄存器地址和STC89xx系列单片机ISP/IAP控制寄存器地址不同如下：

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
STC11/10xx系列 IAP_DATA STC89xx 系列 ISP_DATA	C2h E2h	ISP/IAP Flash Data Register									1111,1111
STC11/10xx系列 IAP_ADDRH STC89xx 系列 ISP_ADDRH	C3h E3h	ISP/IAP Flash Address High									0000,0000
STC11/10xx系列 IAP_ADDRL STC89xx 系列 ISP_ADDRL	C4h E4h	ISP/IAP Flash Address Low									0000,0000
STC11/10xx系列 IAP_CMD STC89xx 系列 ISP_CMD	C5h E5h	ISP/IAP Flash Command Register	-	-	-	-	-	-	MS1	MS0	xxxx,xx00
STC11/10xx系列 IAP_TRIG STC89xx 系列 ISP_TRIG	C6h E6h	ISP/IAP Flash Command Trigger									xxxx,xxxx
STC11/10xx系列 IAP_CONTR STC89xx 系列 ISP_CONTR	C7h E7h	ISP/IAP Control Register	IAPEN	SWBS	SWRST	CMD_FAIL -	-	WT2	WT1	WT0	0000,x000

ISP/IAP_TRIG 寄存器有效启动 IAP 操作，需顺序送入的数据不一样：

STC11/10xx 系列单片机的 ISP/IAP 命令要生效，要对 IAP_TRIG 寄存器按顺序先送 5Ah，再送 A5h 方可

STC89xx 系列单片机的 ISP/IAP 命令要生效，要对 IAP_TRIG 寄存器按顺序先送 46h，再送 B9h 方可

EEPROM 起始地址不一样：

STC11/10xx 系列单片机的 EEPROM 起始地址全部从 0000h 开始，每个扇区 512 字节

STC89xx 系列单片机的 EEPROM 起始地址分别有从 1000h/2000h/4000h/8000h 开始的，程序兼容性不够好.

外部时钟和内部时钟：

最新 STC11/10xx 系列单片机有内部 R/C 振荡器作为系统时钟，一般情况下，44/40 脚封装单片机出厂时的设置是使用外部时钟，20/18/16 脚封装单片机出厂时的设置是使用内部 R/C 振荡器作为系统时钟，用户可在 ISP 烧录用户程序时任意选择使用内部 R/C 时钟或外部晶体 / 时钟.

传统 STC89 系列单片机只能使用外部晶体或时钟作为系统时钟.

功耗：

功耗由 2 部分组成，晶体振荡器放大电路的功耗和单片机的数字电路功耗组成，

晶体振荡器放大电路的功耗： 最新 STC11/10xx 系列单片机比 STC89xx 系列低.

单片机的数字电路功耗:时钟频率越高，功耗越大，最新STC11/10xx系列单片机在相同工作频率下，指令执行速度比传统STC89系列单片机快3-24倍，故可用较低的时钟频率工作，这样功耗更低.建议低功耗设计系统外接4-6MHz的晶体或用内部R/C振荡器作为系统时钟，并利用内部的时钟分频器对时钟进行分频，以较低的频率工作，这样单片机的功耗更低

掉电唤醒：

最新 STC11/10xx 系列单片机支持外部中断模式是下降沿就下降沿唤醒，是低电平就低电平唤醒，传统 STC89系列单片机是外部中断口只要是低电平就唤醒，另最新STC11xx系列还有内部专用掉电唤醒定时器可唤醒，另外，STC11xx系列掉电唤醒延时时间可选:32768/16384/8192/4096个时钟，STC89系列固定是1024个时钟

第二章　　STC11/10xx 系列单片机的 I/O 口结构

2.1　I/O口各种不同的工作模式及配置介绍

I/O 口配置

STC11/10xx 系列单片机其所有 I/O 口均可由软件配置成 4 种工作类型之一，如下表所示。4 种类型分别为：准双向口（标准 8051 输出模式）、推挽输出、仅为输入（高阻）或开漏输出功能。每个口由 2 个控制寄存器中的相应位控制每个引脚工作类型。STC11/10xx 系列单片机上电复位后为准双向口（传统 8051 的 I/O 口）模式。2V 以上时为高电平，0.8V 以下时为低电平。

I/O 口工作类型设定

P4 口设定 <P4.7，P4.6 ,P4.5,P4.4,P4.3,P4.2,P4.1,P4.0 >

P4M1【7：0】	P4M0【7：0】	I/O 口模式
0	0	准双向口(传统8051 I/O 口模式）， 灌电流可达20mA，拉电流为230μA， 由于制造误差，实际为250uA～150uA
0	1	推挽输出（强上拉输出，可达20mA,要加限流电阻）
1	0	仅为输入（高阻）
1	1	开漏(Open Drain)，内部上拉电阻断开，要外加

P3 口设定 <P3.7，P3.6 ,P3.5,P3.4,P3.3,P3.2,P3.1,P3.0 >

P3M1【7：0】	P3M0【7：0】	I/O 口模式
0	0	准双向口(传统8051 I/O 口模式）， 灌电流可达20mA，拉电流为230μA， 由于制造误差，实际为250uA～150uA
0	1	推挽输出（强上拉输出，可达20mA,要加限流电阻）
1	0	仅为输入（高阻）
1	1	开漏(Open Drain)，内部上拉电阻断开，要外加

P2 口设定 <P2.7,P2.6,P2.5,P2.4,P2.3,P2.2,P2.1,P2.0>

P2M1【7：0】	P2M0【7：0】	I/O 口模式
0	0	准双向口(传统8051 I/O 口模式）， 灌电流可达20mA，拉电流为230μA， 由于制造误差，实际为250uA～150uA
0	1	推挽输出（强上拉输出，可达20mA，要加限流电阻）
1	0	仅为输入（高阻）
1	1	开漏(Open Drain)，内部上拉电阻断开，要外加

P1 口设定 <P1.7,P1.6,P1.5,P1.4,P1.3,P1.2,P1.1,P1.0>

P1M1【7：0】	P1M0【7：0】	I/O 口模式（P1.x 如做A/D使用，需先将其设置成开漏或高阻输入）
0	0	准双向口（传统8051 I/O 口模式）， 灌电流可达20mA，拉电流为230μA， 由于制造误差，实际为250uA～150uA
0	1	推挽输出（强上拉输出，可达20mA，要加限流电阻）
1	0	仅为输入（高阻），如果该I/O口需作为A/D使用，可选此模式
1	1	开漏(Open Drain)，如果该I/O口需作为A/D使用，可选此模式

P0 口设定 < P0.7 , P0.6 , P0.5 , P0.4 , P0.3, P0.2, P0.1, P0.0 口>

P0M1【7：0】	P0M0【7：0】	I/O 口模式
0	0	准双向口(传统8051 I/O 口模式）， 灌电流可达20mA，拉电流为230μA， 由于制造误差，实际为250uA～150uA
0	1	推挽输出（强上拉输出，可达20mA，要加限流电阻）
1	0	仅为输入（高阻）
1	1	开漏(Open Drain)，内部上拉电阻断开，要外加

举例：MOV　P1M1，　#10100000B
　　MOV　P1M0，　#11000000B
　　;P1.7 为开漏,P1.6 为强推挽输出,P1.5 为高阻输入,P1.4/P1.3/P1.2/P1.1/P1.0 为弱上拉

注意：

虽然每个 I/O 口在弱上拉时都能承受 20mA 的灌电流(还是要加限流电阻,如 1K，560 Ω等),在强推挽输出时都能输出 20mA 的拉电流（也要加限流电阻),但整个芯片的工作电流推荐不要超过 55mA。即从 MCU-VCC 流入的电流不超过 55mA,从 MCU-Gnd 流出电流不超过 55mA,整体流入 / 流出电流都不能超过 55mA.

STC11/10xx 系列 8051 单片机 I/O 口 特殊功能寄存器　Port SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
P0	80h	8-bit Port 0	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	1111,1111
P0M1	93h										0000,0000
P0M0	94h										0000,0000
P1	90h	8-bit Port 1	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0	1111,1111
P1M1	91h										0000,0000
P1M0	92h										0000,0000
P2	A0h	8-bit Port 2	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	1111,1111
P2M1	95h										0000,0000
P2M0	96h										0000,0000
P3	B0h	8-bit Port 3	P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0	1111,1111
P3M1	B1h										0000,0000
P3M0	B2h										0000,0000
P4	C0h	8-bit Port 3	P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0	1111,1111
P4M1	B3h										0000,0000
P4M0	B4h										0000,0000
P4SW	BBh	Port 4 Switch	-	NA_P4.6	ALE_P4.5	NA_P4.4	-	-	-	-	x000,xxxx

2.2 I/O口各种不同的工作模式结构框图

1.准双向口输出配置

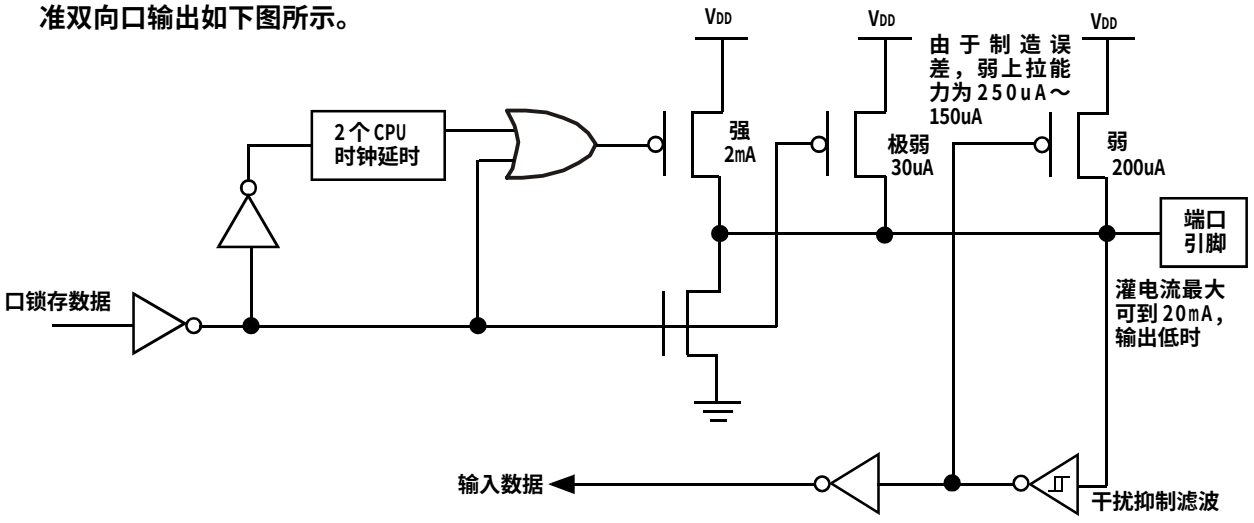
准双向口输出类型可用作输出和输入功能而不需重新配置口线输出状态。这是因为当口线输出为 1 时驱动能力很弱，允许外部装置将其拉低。当引脚输出为低时，它的驱动能力很强，可吸收相当大的电流。准双向口有 3 个上拉晶体管适应不同的需要。

在 3 个上拉晶体管中，有 1 个上拉晶体管称为“弱上拉”，当口线寄存器为 1 且引脚本身也为 1 时打开。此上拉提供基本驱动电流使准双向口输出为 1。如果一个引脚输出为 1 而由外部装置下拉到低时，弱上拉关闭而“极弱上拉”维持开状态，为了把这个引脚强拉为低，外部装置必须有足够的灌电流能力使引脚上的电压降到门槛电压以下。

第 2 个上拉晶体管，称为“极弱上拉”，当口线锁存为 1 时打开。当引脚悬空时，这个极弱的上拉源产生很弱的上拉电流将引脚上拉为高电平。

第 3 个上拉晶体管称为“强上拉”。当口线锁存器由 0 到 1 跳变时，这个上拉用来加快准双向口由逻辑 0 到逻辑 1 转换。当发生这种情况时，强上拉打开约 2 个时钟以使引脚能够迅速地上拉到高电平。

准双向口输出如下图所示。



STC11/10Lxx 系列单片机为 3V 器件，如果用户在引脚加上 5V 电压，将会有电流从引脚流向 VDD，这样导致额外的功率消耗。因此，建议不要在准双向口模式中向 3V 单片机引脚施加 5V 电压,如使用的话，要加限流电阻，或用二极管做输入隔离，或用三极管做输出隔离。

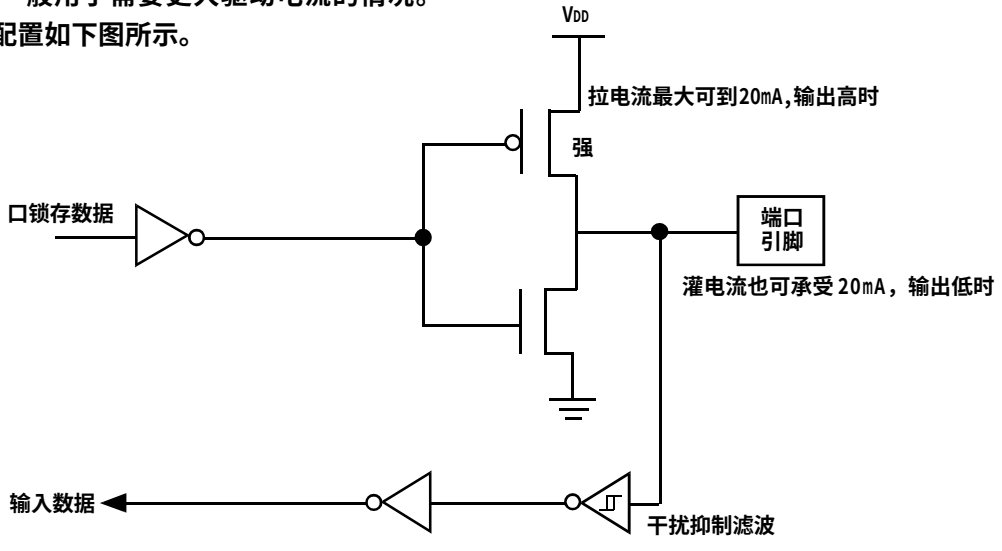
准双向口带有一个施密特触发输入以及一个干扰抑制电路。

准双向口读外部状态前,要先锁存为 ‘1’,才可读到外部正确的状态.

2.推挽输出配置

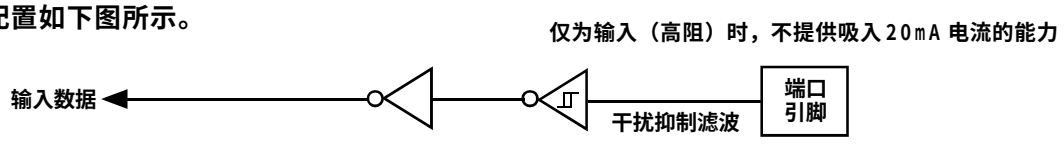
推挽输出配置的下拉结构与开漏输出以及准双向口的下拉结构相同，但当锁存器为 1 时提供持续的强上拉。推挽模式一般用于需要更大驱动电流的情况。

推挽引脚配置如下图所示。



3．仅为输入（高阻）配置

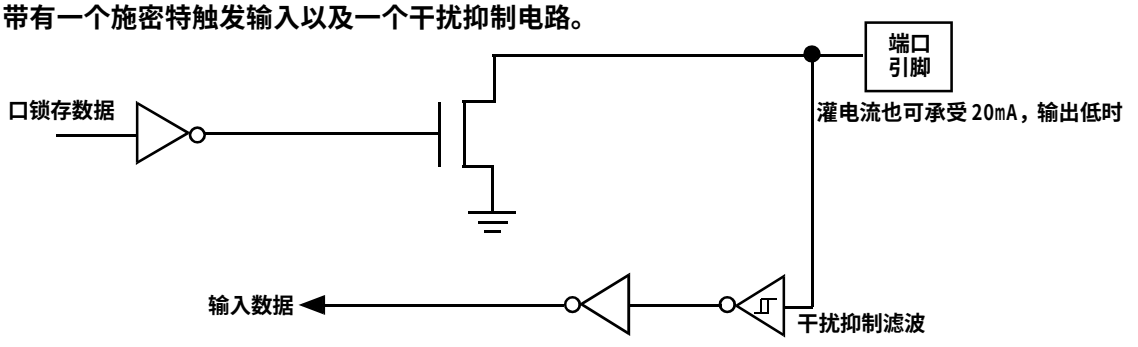
输入口配置如下图所示。



输入口带有一个施密特触发输入以及一个干扰抑制电路。

4.开漏输出配置

当口线锁存器为0时，开漏输出关闭所有上拉晶体管。当作为一个逻辑输出时，这种配置方式必须有外部上拉，一般通过电阻外接到VDD。这种方式的下拉与准双向口相同。输出口线配置如下图所示。
开漏端口带有一个施密特触发输入以及一个干扰抑制电路。



关于I/O口应用注意事项：

少数用户反映I/O口有损坏现象,后发现是

有些是I/O口由低变高读外部状态时,读不对,实际没有损坏,软件处理一下即可

是因为1T的8051单片机速度太快了,软件执行由低变高指令后立即读外部状态,此时由于实际输出还没有变高,就有可能读不对,正确的方法是在软件设置由低变高后加1到2个空操作指令延时,再读就对了.

有些实际没有损坏,加上拉电阻就OK了

是因为外围接的是SPI/I2C等漏极开漏的电路,要加10K上拉电阻.

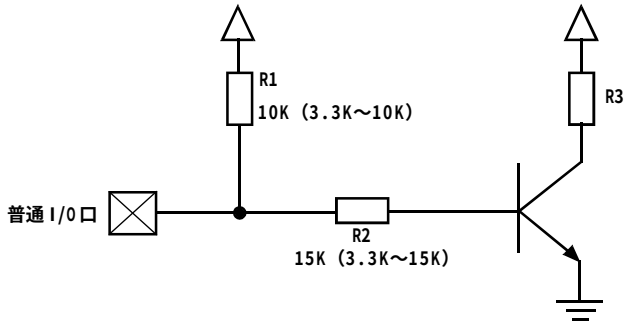
有些是外围接的是NPN三极管,没有加上拉电阻,其实基极串多大电阻,I/O口就应该上拉多大的电阻,或者将该I/O口设置为强推挽输出.

有些确实是损坏了,原因:

发现有些是驱动LED发光二极管没有加限流电阻,建议加1K以上的限流电阻,至少也要加470欧姆以上

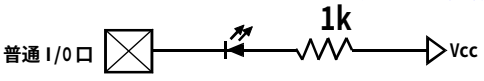
发现有些是做行列矩阵按键扫描电路时,实际工作时没有加限流电阻,实际工作时可能出现2个I/O口均输出为低,并且在按键按下时,短接在一起,我们知道一个CMOS电路的2个输出脚不应该直接短接在一起,按键扫描电路中,此时一个口为了读另外一个口的状态,必须先置高才能读另外一个口的状态,而8051单片机的弱上拉口在由0变为1时,会有2个时钟的强推挽高输出电流,输出到另外一个输出为低的I/O口,就有可能造成I/O口损坏.建议在其中的一侧加1K限流电阻,或者在软件处理上,不要出现按键两端的I/O口同时为低.

2.3 一种典型三极管控制电路



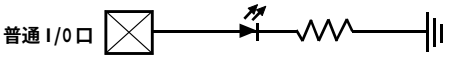
如果用弱上拉控制，建议加上拉电阻R1（3.3K～10K），如果不加上拉电阻R1（3.3K～10K），建议R2的值在15K以上，或用强推挽输出

2.4 典型发光二极管控制电路



弱上拉/准双向口，用灌电流驱动发光二极管

限流电阻尽量大于1K，最小不要小于470欧姆

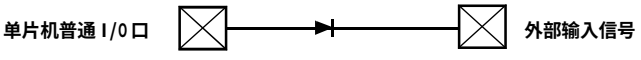


推挽/强上拉口，

2.5 3V/5V混合系统I/O口互连

STC11/10xx系列5V单片机连接3V器件时，为防止3V器件承受不了5V，可将相应的I/O口设置成开漏配置，断开内部上拉电阻，相应的I/O口外部加10K上拉电阻到3V器件的Vcc，这样高电平是3V，低电平是0V，输入输出一切正常。

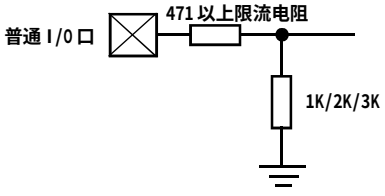
STC11/10xx系列3V单片机连接5V器件时，为防止3V器件承受不了5V，如果相应的I/O口是输入，可在该I/O口上串接一个隔离二极管，隔离高压部分。外部信号电压高于单片机工作电压时截止，I/O口此时已内部上拉到高电平；外部信号电压为低时导通，I/O口被钳位在0.7V，小于0.8V时单片机就认为是低电平。



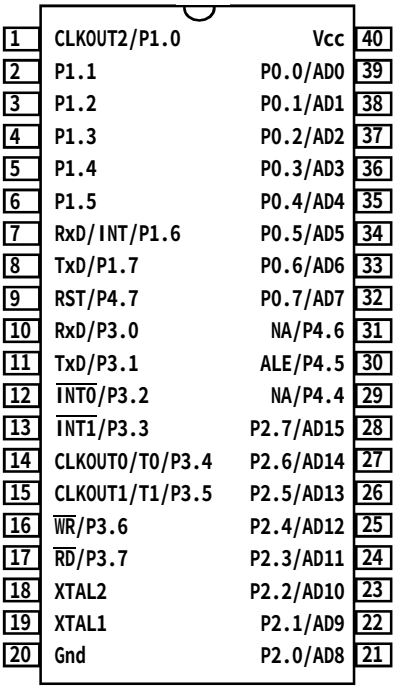
2.6 如何让I/O口上电复位时为低电平

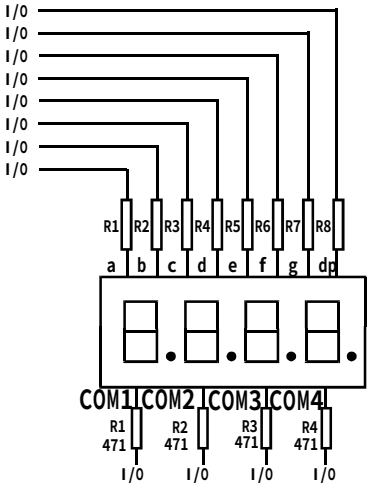
普通8051单片机上电复位时普通I/O口为弱上拉高电平输出,而很多实际应用要求上电时某些I/O口为低电平输出,否则所控制的系统(如马达)就会误动作,现STC12系列单片机由于既有弱上拉输出又有强推挽输出,就可以很轻松的解决此问题。

现可在STC11/10xx系列单片机I/O口上加一个下拉电阻(1K/2K/3K),这样上电复位时，虽然单片机内部I/O口是弱上拉/高电平输出，但由于内部上拉能力有限，而外部下拉电阻又较小,无法将其拉高,所以该I/O口上电复位时外部为低电平。如果要将此I/O口驱动为高电平，可将此I/O口设置为强推挽输出，而强推挽输出时，I/O口驱动电流可达20mA,故肯定可以将该口驱动为高电平输出。



2.7 I/O口驱动LED数码管应用线路图

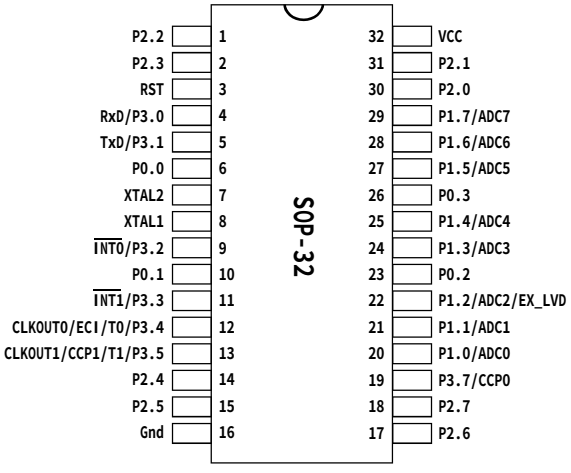


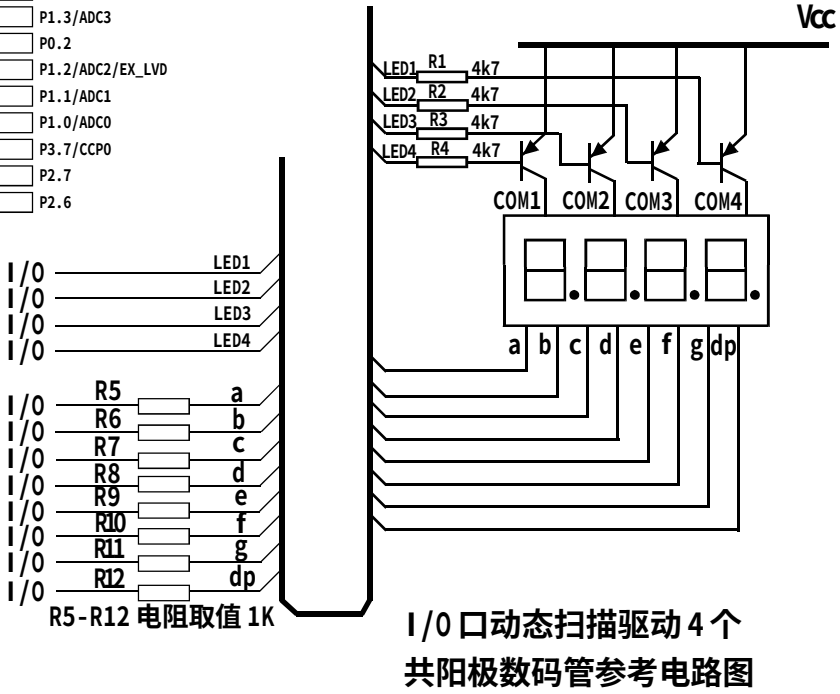


R1-R8 阻值为 470 欧姆

I/O口动态扫描驱动4个共阴极数码管参考电路图

I/O口动态扫描驱动数码管时，可以一次点亮一个数码管中的8段，但为降低功耗，建议可以一次只点亮其中的4段或者2段

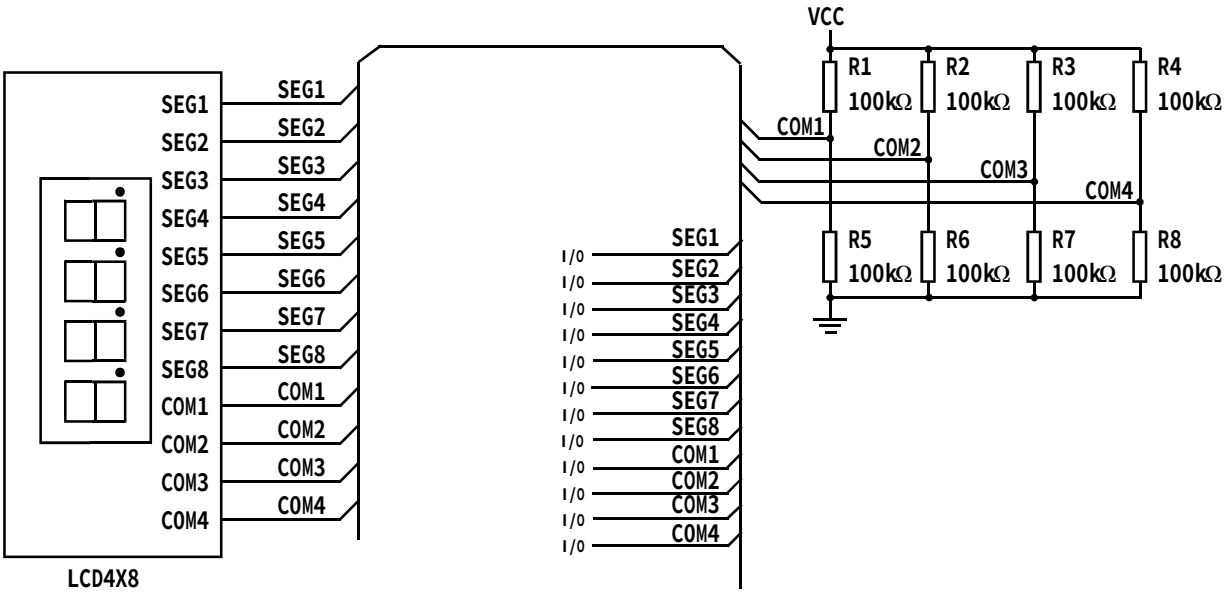




R5-R12 电阻取值 1K

I/O口动态扫描驱动4个共阳极数码管参考电路图

2.8 I/O口直接驱动LCD应用线路图



如何点亮相应的LCD像素：

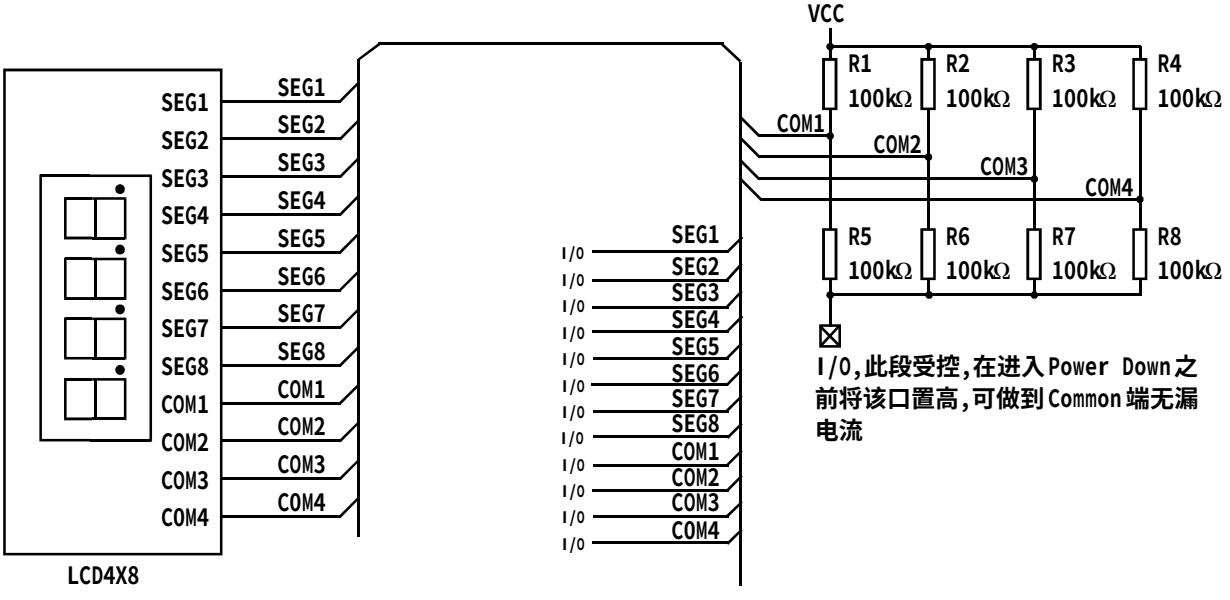
当相应的Common端和相应的Segment端压差大于1/2Vcc时，相应的像素就显示，当压差小于1/2Vcc时，相应的像素就不显示

I/O口如何控制Segment：

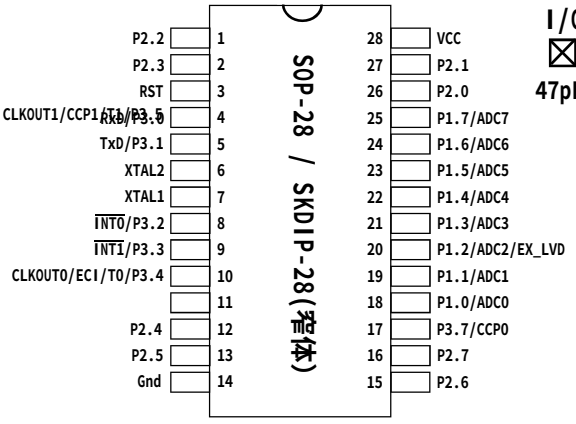
I/O口直接控制Segment，程序控制相应的口输出高或低时，对应的Segment就是Vcc或0V

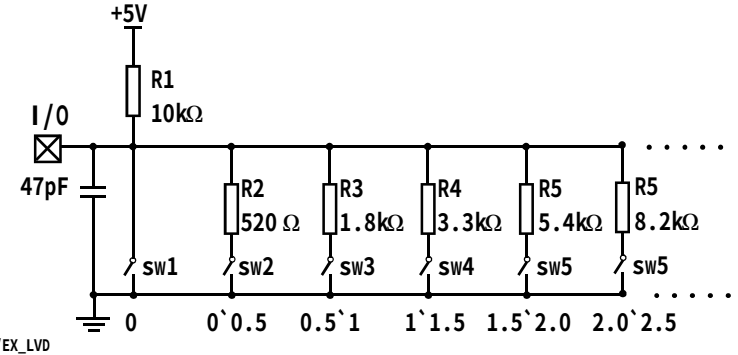
I/O口如何控制Common：

I/O口和2个100K的分压电阻组成Common，当I/O 口输出为0时，相应的Common端为0V，当I/O 口强推挽输出为1时，相应的Common端为Vcc，当I/O 口为高阻输入时，相应的Common端为1/2Vcc，



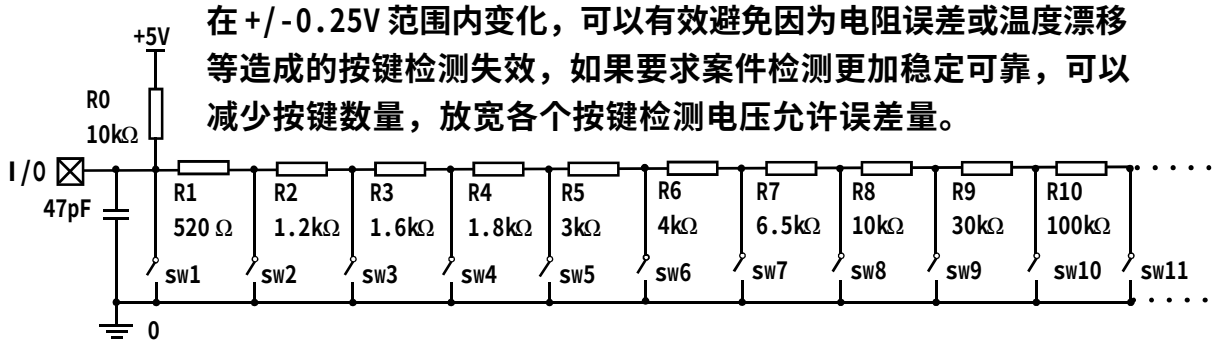
2.9 A/D 做按键扫描应用线路图





此电路可以实现单个按键扫描和组合按键检测功能,但是具体电阻值应根据实际需要进行选择

本电路图采用 10 个按键等间隔分压，每个按键正负误差余量允许在 +/-0.25V 范围内变化，可以有效避免因为电阻误差或温度漂移等造成的按键检测失效，如果要求案件检测更加稳定可靠，可以减少按键数量，放宽各个按键检测电压允许误差量。



第三章　STC11Fxx系列单片机看门狗应用及软件复位

3.1　看门狗应用及测试程序

3.1.1　看门狗应用介绍

适用型号：　STC11/10xx 系列

在工业控制 / 汽车电子 / 航空航天等需要高可靠性的系统中,为了防止“系统在异常情况下，受到干扰，MCU/CPU 程序跑飞，导致系统长时间异常工作”,通常是引进看门狗,如果 MCU/CPU 不在规定的时间内按要求访问看门狗,就认为MCU/CPU处于异常状态,看门狗就会强迫MCU/CPU复位,使系统重新从头开始按规律执行用户程序。STC11Fxx系列单片机内部也引进了此看门狗功能，使单片机系统可靠性设计变得更加方便/简洁。为此功能，我们增加如下特殊功能寄存器 WDT_CONTR：

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WDT_CONTR	C1h	Watch-Dog-Timer Control register	WDT_FLAG	-	EN_WDT	CLR_WDT	IDLE_WDT	PS2	PS1	PS0	xx00,0000

Symbol 符号 Function 功能

WDT_FLAG　When WDT overflows, this bit is set. It can be cleared by software.
看门狗溢出标志位,当溢出时，该位由硬件置 1，可用软件将其清 0。

EN_WDT　Enable WDT bit. When set, WDT is started
看门狗允许位，　当设置为“ 1 ”时，看门狗启动。

CLR_WDT　WDT clear bit. When set, WDT will recount. Hardware will automatically clear this bit.
看门狗清“ 0 ”位，当设为“ 1 ”时，看门狗将重新计数。硬件将自动清“ 0 ”此位。

IDLE_WDT When set, WDT is enabled in IDLE mode. When clear, WDT is disabled in IDLE
看门狗“ IDLE ”模式位，当设置为“ 1 ”时，看门狗定时器在“ 空闲模式 ”计数
当清“ 0 ”该位时，看门狗定时器在“ 空闲模式 ”时不计数

PS2, PS1, PS0 Pre-scale value of Watchdog timer is shown as the bellowed table:
看门狗定时器预分频值，如下表所示

PS2	PS1	PS0	Pre-scale 预分频	WDT Period @20MHz
0	0	0	2	39.3 mS
0	0	1	4	78.6 mS
0	1	0	8	157.3 mS
0	1	1	16	314.6 mS
1	0	0	32	629.1 mS
1	0	1	64	1.25S
1	1	0	128	2.5S
1	1	1	256	5S

The WDT period is determined by the following equation 看门狗溢出时间计算
看门狗溢出时间 = (12 x Pre-scale x 32768) / Oscillator frequency
设时钟为 12MHz：
看门狗溢出时间 = （

PS2	PS1	PS0	Pre-scale 预分频	WDT Period @12MHz
0	0	0	2	65.5 mS
0	0	1	4	131.0 mS
0	1	0	8	262.1 mS
0	1	1	16	524.2 mS
1	0	0	32	1.0485S
1	0	1	64	2.0971S
1	1	0	128	4.1943S
1	1	1	256	8.3886S

设时钟为 11.0592MHz：

看门狗溢出时间 = (12 x Pre-scale x 32768) / 11059200 = Pre-scale x 393216 / 11059200

PS2	PS1	PS0	Pre-scale 预分频	WDT Period @11.0592MHz
0	0	0	2	71.1 mS
0	0	1	4	142.2 mS
0	1	0	8	284.4 mS
0	1	1	16	568.8 mS
1	0	0	32	1.1377S
1	0	1	64	2.2755S
1	1	0	128	4.5511S
1	1	1	256	9.1022S

汇编语言程序示例

```
WDT_CONTR   DATA0C1H ;    或者   WDT_CONTR   EQU   0C1H
;复位入口
    ORG     0000H
    LJMP    Initial
    …
    ORG     0060H
Initial:
    MOV     WDT_CONTR,  #00111100B; Load initial value 看门狗定时器控制寄存器初始化
              ; EN_WDT = 1, CLR_WDT = 1, IDLE_WDT = 1, PS2 = 1, PS1 = 0, PS0 = 0
    …
Main_Loop:
    LCALL   Display_Loop
    LCALL   Keyboard_Loop
    …
    MOV     WDT_CONTR,  #00111100B ;  喂狗，  不要用 ORL    WDT_CONTR,  #00010000B
    …
    LJMP    Main_Loop
```

C 语言程序示例

```
#include<reg52.h>
sfr     WDT_CONTR   =   0xc1;
void main()
{   …
    WDT_CONTR   =   0x3c;
       /* 0011,1100 EN_WDT = 1,CLR_WDT = 1,IDLE_WDT = 1,PS2 = 1,PS1 = 0,PS0 = 0 */
    while(1){
        display();
        keyboard();
        …
        WDT_CONTR   =   0x3c;  /* 喂狗，  不要用  WDT_CONTR   =   WDT_CONTR | 0x10;*/
    }
}
```

3.1.2 一个完整的看门狗测试程序，在宏晶的下载板上可以直接测试

本程序验证 STC11/10xx 系列单片机的看门狗及其溢出时间计算公式

```
;/* ---  STC International Limited ------------------------------------------------- */

;/* ---  演示 STC11/10xx 系列 MCU 看门狗及其溢出时间计算公式 ---------------------------- */

;如果要在程序中使用或在文章中引用该程序,请在程序中或文章中注明使用了宏晶科技的资料及程序
;本演示程序在 STC-ISP Ver 3.0A.PCB 的下载编程工具上测试通过,相关的工作状态在 P1 口上显示
;看门狗及其溢出时间 = (12 * Pre_scale *32768)/Oscillator frequency
WDT_CONTR       EQU 0C1H ;看门狗地址
WDT_TIME_LED    EQU P1.5 ;用 P1.5 控制看门狗溢出时间指示灯,
                         ;看门狗溢出时间可由该指示灯亮的时间长度或熄灭的时间长度表示
WDT_FLAG_LED    EQU P1.7 ;用 P1.7 控制看门狗溢出复位指示灯，如点亮表示为看门狗溢出复位
Last_WDT_Time_LED_Status  EQU 00H ;位变量，存储看门狗溢出时间指示灯的上一次状态位
;WDT 复位时间(所用的 Oscillator frequency = 18.432MHz):
;Pre_scale_Word  EQU 00111100B   ;清 0,启动看门狗，预分频数 =32,   0.68S
Pre_scale_Word  EQU 00111101B    ;清 0,启动看门狗，预分频数 =64,   1.36S
;Pre_scale_Word  EQU 00111110B   ;清 0,启动看门狗，预分频数 =128,  2.72S
;Pre_scale_Word  EQU 00111111B   ;清 0,启动看门狗，预分频数 =256,  5.44S
    ORG   0000H
    AJMP  MAIN
    ORG   0100H
MAIN:
    MOV   A, WDT_CONTR          ;检测是否为看门狗复位
    ANL   A, #10000000B
    JNZ   WDT_Reset             ;WDT_CONTR.7 = 1，看门狗复位，跳转到看门狗复位程序
;WDT_CONTR.7 = 0,上电复位，冷启动，RAM 单元内容为随机值
    SETB  Last_WDT_Time_LED_Status       ;上电复位,
                                         ;初始化看门狗溢出时间指示灯的状态位 = 1
    CLR   WDT_TIME_LED                   ;上电复位，点亮看门狗溢出时间指示灯
    MOV   WDT_CONTR, #Pre_scale_Word     ;启动看门狗
WAIT1:
    SJMP  WAIT1                  ;循环执行本语句(停机)，等待看门狗溢出复位

;WDT_CONTR.7 = 1,看门狗复位，热启动，RAM 单元内容不变，为复位前的值
WDT_Reset:                       ;看门狗复位，热启动
    CLR   WDT_FLAG_LED           ;是看门狗复位,点亮看门狗溢出复位指示灯

    JB    Last_WDT_Time_LED_Status, Power_Off_WDT_TIME_LED;为 1 熄灭相应的灯,为 0 亮相应灯
    ;根据看门狗溢出时间指示灯的上一次状态位设置 WDT_TIME_LED 灯,
    ;若上次亮本次就熄灭，若上次熄灭本次就亮
```

```
    CLR   WDT_TIME_LED              ;上次熄灭本次点亮看门狗溢出时间指示灯
    CPL   Last_WDT_Time_LED_Status ;将看门狗溢出时间指示灯的上一次状态位取反
WAIT2:
    SJMP  WAIT2                     ;循环执行本语句(停机)，等待看门狗溢出复位
Power_Off_WDT_TIME_LED:
    SETB  WDT_TIME_LED              ;上次亮本次就熄灭看门狗溢出时间指示灯
    CPL   Last_WDT_Time_LED_Status ;将看门狗溢出时间指示灯的上一次状态位取反
WAIT3:
    SJMP  WAIT3                     ;循环执行本语句(停机)，等待看门狗溢出复位
    END
```

3.2 如何用软件实现系统复位

用户应用程序在运行过程当中，有时会有特殊需求，需要实现单片机系统软复位（热启动之一），传统的 8051 单片机由于硬件上未支持此功能，用户必须用软件模拟实现，实现起来较麻烦。现 STC 新推出的增强型 8051 根据客户要求增加了 IAP_CONTR 特殊功能寄存器,实现了此功能。用户只需简单的控制 IAP_CONTR 特殊功能寄存器的其中两位 SWBS / SWRST 就可以系统复位了。

IAP_CONTR: IAP 控制寄存器，地址在 0C7H 单元

B7	B6	B5	B4	B3	B2	B1	B0	Reset Value
IAPEN	SWBS	SWRST	CMD_FAIL	-	WT2	WT1	WT0	0000,x000

IAPEN: ISP/IAP 功能允许位。0：禁止 IAP 编程改变 Flash,1:允许编程改变 Flash

SWBS: 软件选择从用户应用程序区启动（0），还是从 ISP 程序区启动（1）。要与 SWRST 直接配合才可以实现

SWRST: 0：不操作； 1： 产生软件系统复位，硬件自动清零。

CMD_FAIL: 如果送了 ISP/IAP 命令，并对 IAP_TRIG 送 5Ah/A5h 触发失败，则为 1，需由软件清零。

```
;从用户应用程序区(AP 区)软件复位并切换到用户应用程序区(AP 区)开始执行程序
    MOV   IAP_CONTR, #00100000B ;SWBS = 0(选择 AP 区), SWRST = 1(软复位)
;从系统 ISP 监控程序区软件复位并切换到用户应用程序区(AP 区)开始执行程序
    MOV   IAP_CONTR, #00100000B ;SWBS = 0(选择 AP 区), SWRST = 1(软复位)

;从用户应用程序区(AP 区)软件复位并切换到系统 ISP 监控程序区开始执行程序
    MOV   IAP_CONTR, #01100000B ;SWBS = 1(选择 ISP 区), SWRST = 1(软复位)
;从系统 ISP 监控程序区软件复位并切换到系统 ISP 监控程序区开始执行程序
    MOV   IAP_CONTR, #01100000B ;SWBS = 1(选择 ISP 区), SWRST = 1(软复位)
```

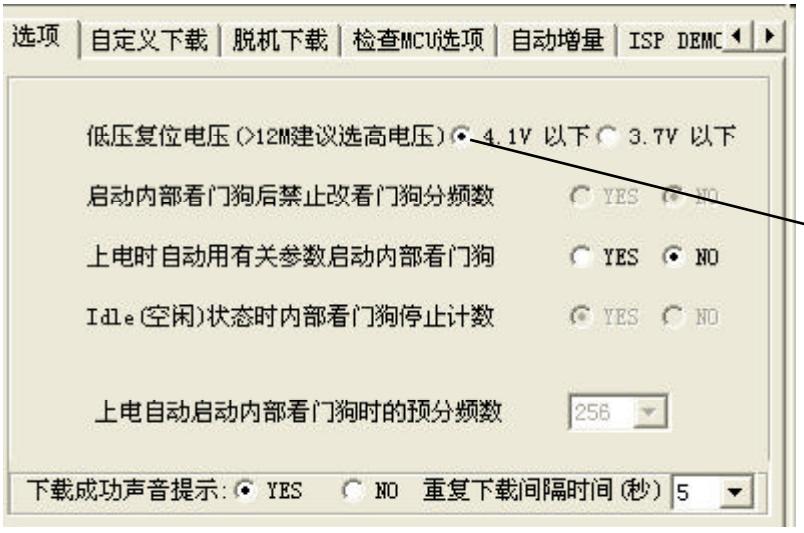
本复位是整个系统复位，所有的特殊功能寄存器都会复位到初始值，I/O 口也会初始化。

4.3 热启动复位和冷启动复位

	复位源	现象
热启动复位	内部看门狗复位	会使单片机直接从用户程序区 0000H 处开始执行用户程序
	通过控制 RESET 脚产生的硬复位	会使系统从用户程序区 0000H 处开始直接执行用户程序
	通过对 IAP_CONTR 寄存器送入 20H 产生的软复位	会使系统从用户程序区 0000H 处开始直接执行用户程序
	通过对 IAP_CONTR 寄存器送入 60H 产生的软复位	会使系统从系统 ISP 监控程序区开始执行程序，检测不到合法的 ISP 下载命令流后，会软复位到用户程序区执行用户程序
冷启动复位	系统停电后再上电引起的硬复位	会使系统从系统 ISP 监控程序区开始执行程序，检测不到合法的 ISP 下载命令流后，会软复位到用户程序区执行用户程序

3.4 复位门槛电压选择

STC11/10xx 系列单片机都有 2 档复位门槛电压供用户选择



STC11Fxx 系列单片机
复位门槛电压选择:

STC11Fxx 系列 5V 单片机:

晶振频率在 24M 以下时，选择 4.1V 以下复位，系统复位才可靠

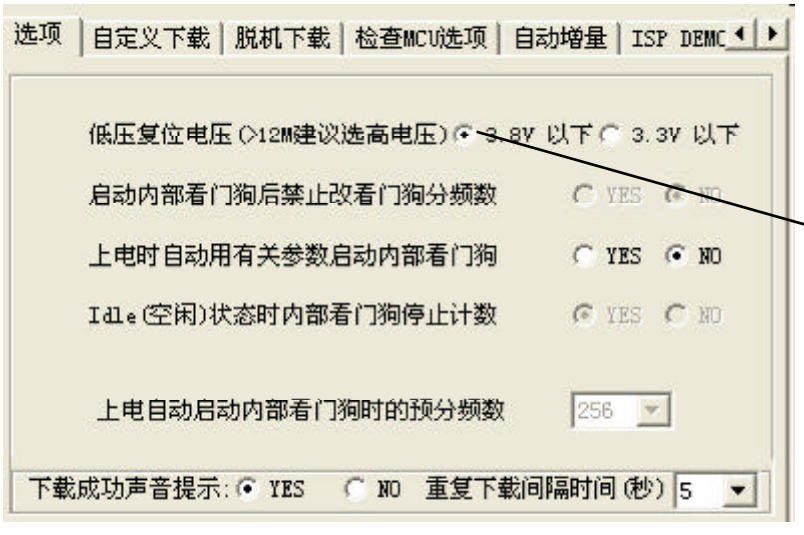
晶振频率在 12M 以下时，可以选择 4.1V 以下复位,也可以选择 3.7V 以下复位

但 STC11F01/02/03/04/05,STC11F01E/02E/03E/04E/05E,IAP11F06 的复位门槛电压为 4.1V 或 3.5V 可选

STC11Lxx 系列 3V 单片机:

晶振频率在 24M 以下时，选择 2.4V 以下复位，系统复位才可靠

晶振频率在 12M 以下时，可以选择 2.4V 以下复位,也可以选择 2.1V 以下复位



STC10Fxx 系列单片机
复位门槛电压选择

STC10Fxx 系列 5V 单片机：

晶振频率在 20M 以下时，选择 3.8V 以下复位，系统复位才可靠

晶振频率在 12M 以下时，可以选择 3.8V 以下复位,也可以选择 3.3V 以下复位

STC10Lxx 系列 3V 单片机：

晶振频率在 24M 以下时，选择 2.4V 以下复位，系统复位才可靠

晶振频率在 12M 以下时，可以选择 2.4V 以下复位,也可以选择 2.1V 以下复位

第四章　STC11/10xx 系列单片机 EEPROM 的应用

--- 利用 ISP/IAP 技术将内部 Data Flash 当 EEPROM，擦写次数 10 万次以上

4.1　IAP 及 EEPROM 新增特殊功能寄存器介绍及使用

STC11xx 系列 5V 单片机在 4.1V 以上对 EEPROM 进行操作才有效,4.1V 以下对 EEPROM 进行操作,MCU 不执行此功能;3.3V 单片机在 2.4V 以上对 EEPROM 进行操作才有效,2.4V 以下对 EEPROM 操作,MCU 不执行此功能。

STC10xx 系列 5V 单片机在 3.8V 以上对 EEPROM 进行操作才有效,3.8V 以下对 EEPROM 进行操作,MCU 不执行此功能;3.3V单片机在2.4V以上对EEPROM进行操作才有效,2.4V以下对EEPROM操作,MCU 不执行,所以建议ISP烧录程序时选择高的复位门槛电压，如 5V 单片机选择 4.1V 以下复位，3V 单片机选择 2.4V 以下复位。

STC11/10xx 系列 1T 8051 单片机 ISP/IAP 特殊功能寄存器　　ISP/IAP SFRs

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
IAP_DATA	C2h	ISP/IAP Flash Data Register									1111,1111
IAP_ADDRH	C3h	ISP/IAP Flash Address High									0000,0000
IAP_ADDRL	C4h	ISP/IAP Flash Address Low									0000,0000
IAP_CMD	C5h	ISP/IAP Flash Command Register	-	-	-	-	-	-	MS1	MS0	xxxx,xx00
IAP_TRIG	C6h	ISP/IAP Flash Command Trigger									xxxx,xxxx
IAP_CONTR	C7h	ISP/IAP Control Register	IAPEN	SWBS	SWRST	CMD_FAIL	-	WT2	WT1	WT0	0000,x000

IAP_DATA:　　ISP/IAP 操作时的数据寄存器。

ISP/IAP 从 Flash 读出的数据放在此处，向 Flash 写的数据也需放在此处

IAP_ADDRH:　　ISP/IAP 操作时的地址寄存器高八位。

IAP_ADDRL:　　ISP/IAP 操作时的地址寄存器低八位。

IAP_CMD:　　ISP/IAP 操作时的命令模式寄存器，须命令触发寄存器触发方可生效。

B7	B6	B5	B4	B3	B2	B1	B0	命令 / 操作 模式选择
保留						命令		
-	-	-	-	-	-	0	0	Standby 待机模式，无ISP/IAP操作
-	-	-	-	-	-	0	1	对 "Data Flash/EEPROM区"进行字节读
-	-	-	-	-	-	1	0	对 "Data Flash/EEPROM区"进行字节编程
-	-	-	-	-	-	1	1	对 "Data Flash/EEPROM区"进行扇区擦除

IAP_TRIG:　ISP/IAP 操作时的命令触发寄存器。

在 IAPEN(IAP_CONTR.7) = 1 时,对 IAP_TRIG 先写入 5Ah,再写入 A5h,ISP/IAP 命令才会生效。

IAP_CONTR: ISP/IAP 控制寄存器，地址在 0C7H 单元

B7	B6	B5	B4	B3	B2	B1	B0	Reset Value
IAPEN	SWBS	SWRST	CMD_FAIL	1	WT2	WT1	WT0	0000,1000

IAPEN:　　ISP/IAP 功能允许位。0：禁止 ISP/IAP 编程改变 Flash,1:允许编程改变 Flash

SWBS:　　软件选择从用户主程序区启动（0），还是从 ISP 程序区启动（1）。

SWRST:　　0：不操作；　　1：产生软件系统复位，硬件自动清零。

CMD_FAIL:　如果送了 ISP/IAP 命令，并对 ISP_TRIG 送 5Ah/A5h 触发失败，则为 1，需由软件清零。

```
;在用户应用程序区(AP 区)软件复位并从用户应用程序区(AP 区)开始执行程序
      MOV   IAP_CONTR, #00100000B ;SWBS = 0(选择 AP 区), SWRST = 1(软复位)
;在用户应用程序区(AP 区)软件复位并从系统 ISP 监控程序区开始执行程序
      MOV   IAP_CONTR, #01100000B ;SWBS = 1(选择 ISP 区), SWRST = 1(软复位)
;在系统 ISP 监控程序区软件复位并从用户应用程序区(AP 区)开始执行程序
      MOV   IAP_CONTR, #00100000B ;SWBS = 0(选择 AP 区), SWRST = 1(软复位)
;在系统 ISP 监控程序区软件复位并从系统 ISP 监控程序区开始执行程序
      MOV   IAP_CONTR, #01100000B ;SWBS = 1(选择 ISP 区), SWRST = 1(软复位)
```

设置等待时间			CPU 等待时间(多少个 CPU 工作时钟)			
WT2	WT1	WT0	Read/读	Program/编程	Sector Erase 扇区擦除	Recommended System Clock 推荐跟等待参数对应的系统时钟
1	1	1	2个时钟	55个时钟 编程一个字节需要55uS	21012个时钟 擦除一个扇区需要21mS	〈 1MHz
1	1	0	2个时钟	110个时钟	42024个时钟	〈 2MHz
1	0	1	2个时钟	165个时钟	63036个时钟	〈 3MHz
1	0	0	2个时钟	330个时钟	126072个时钟	〈 6MHz
0	1	1	2个时钟	660个时钟	252144个时钟	〈 12MHz
0	1	0	2个时钟	1100个时钟	420240个时钟	〈 20MHz
0	0	1	2个时钟	1320个时钟	504288个时钟	〈 24MHz
0	0	0	2个时钟	1760个时钟	672384个时钟	〈 30MHz

EEPROM 使用注意事项:

为了保证单片机内部 EEPROM 的正常可靠工作，目前供货的单片机：

STC11xx 系列：

5V 单片机在 Vcc<4.1V 时，禁止 ISP/IAP 操作，即禁止对 EEPROM 的正常操作，此时单片机对相应的 ISP/IAP 指令不响应，实际情况是，对 ISP/IAP 寄存器的操作是执行了，但由于此时工作电压低于可靠的门槛电压以下，单片机内部此时禁止执行 ISP/IAP 操作，即对 EEPROM 的擦除 / 编程 / 读命令均无效。

3V 单片机在 Vcc<2.4V 时，禁止 ISP/IAP 操作，即禁止对 EEPROM 的正常操作，此时单片机对相应的 ISP/IAP 指令不响应，实际情况是，对 ISP/IAP 寄存器的操作是执行了，但由于此时工作电压低于可靠的门槛电压以下，单片机内部此时禁止执行 ISP/IAP 操作，即对 EEPROM 的擦除 / 编程 / 读命令均无效。

STC10xx 系列：

5V 单片机在 Vcc<3.8V 时，禁止 ISP/IAP 操作，即禁止对 EEPROM 的正常操作，此时单片机对相应的 ISP/IAP 指令不响应，实际情况是，对 ISP/IAP 寄存器的操作是执行了，但由于此时工作电压低于可靠的门槛电压以下，单片机内部此时禁止执行 ISP/IAP 操作，即对 EEPROM 的擦除 / 编程 / 读命令均无效。

3V 单片机在 Vcc<2.4V 时，禁止 ISP/IAP 操作，即禁止对 EEPROM 的正常操作，此时单片机对相应的 ISP/IAP 指令不响应，实际情况是，对 ISP/IAP 寄存器的操作是执行了，但由于此时工作电压低于可靠的门槛电压以下，单片机内部此时禁止执行 ISP/IAP 操作，即对 EEPROM 的擦除 / 编程 / 读命令均无效。

如果电源上电缓慢，可能会由于程序已经开始运行，而此时电源电压还达不到 EEPROM 的最低可靠工作电压，导致执行相应的 EEPROM 指令无效，所以建议用户选择高的复位门槛电压,如果用户需要宽的工作电压范围,选择了低的复位门槛电压复位,建议对 EEPROM 进行操作时，要判断低电压 /LVDF 标志位（PCON 电源管理寄存器的 Bit5 位)。如果该位为“ 1 ”，则说明电源电压曾经低于有效的门槛电压，软件将其清零,加几个空操作延时后再读该位的状态,如果为 ”0 ”，说明工作电压高于有效的门槛电压之上,则可进行 ISP/IAP/EEPROM 操作。如果为 ” 1 ”,则将其再清零，一直等到工作电压高于有效的门槛电压之上，才可进行 ISP/IAP/EEPROM 操作。

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset value
PCON	87h	Power Control	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000

4.2 STC11/10xx系列单片机内部EEPROM大小和地址

STC11/10xx系列单片机内部可用EEPROM的地址与程序空间是分开的：程序在用户应用程序区时，可以对EEPROM进行IAP/ISP操作。

具体某个型号单片机内部EEPROM大小及详细地址请参阅：

1．STC11/10xx系列单片机内部EEPROM详细地址表

2．STC11/10xx系列单片机内部EEPROM空间大小选型一览表

举例如下：

STC11F01E，有2K 字节的EEPROM，则由4个扇区组成，每个扇区0.5K字节，地址从0000h - 0FFFh

STC11F16XE，有32K 字节的EEPROM，则由64个扇区组成，每个扇区0.5K字节，地址从

型号命名为IAP11xx或IAP10xx的单片机，没有独立的EEPROM，但是可利用IAP寄存器在程序区对程序Flash进行直接修改，做自己的在线编程/远程升级，当然，也可将程序Flash当EEPROM使用.扇区地址参照STC11/10xx系列单片机EEPROM详细地址表

STC11Fxx/STC11Lxx系列单片机内部EEPROM空间大小选型一览表				
型号	EEPROM字节数	扇区数	起始扇区起始地址	结束扇区结束地址
STC11F01E/STC11L01E	2K	4	0000h	0FFFh
STC11F02E/STC11L02E	2K	4	0000h	0FFFh
STC11F03E/STC11L03E	2K	4	0000h	0FFFh
STC11F04E/STC11L04E	1K	2	0000h	3FFh
STC11F05E/STC11L05E	1K	2	0000h	3FFh
STC11Fxx/STC11Lxx系列单片机内部EEPROM空间大小选型一览表				
STC11F08XE/STC11L08XE	32K	64	0000h	7FFFh
STC11F16XE/STC11L16XE	32K	64	0000h	7FFFh
STC11F20XE/STC11L20XE	29K	58	0000h	73FFh
STC11F32XE/STC11L32XE	29K	58	0000h	73FFh
STC11F40XE/STC11L40XE	21K	42	0000h	53FFh
STC11F48XE/STC11L48XE	13K	26	0000h	33FFh
STC11F52XE/STC11L52XE	9K	18	0000h	23FFh
STC11F56XE/STC11L56XE	5K	10	0000h	13FFh
STC11F60XE/STC11L60XE	1K	2	0000h	3FFh
STC10Fxx系列单片机内部EEPROM空间大小选型一览表				
型号	EEPROM字节数	扇区数	起始扇区起始地址	结束扇区结束地址
STC10F02XE/STC10L02XE	5K	10	0000h	13FFh
STC10F04XE/STC10L04XE	5K	10	0000h	13FFh
STC10F06XE/STC10L06XE	5K	10	0000h	13FFh
STC10F08XE/STC10L08XE	5K	10	0000h	13FFh
STC10F10XE/STC10L10XE	3K	6	0000h	0BFFh
STC10F12XE/STC10L12XE	1K	2	0000h	3FFh

STC11/10xx系列单片机内部EEPROM详细地址表

具体某型号有多少扇区的EEPROM,参照前面的EEPROM空间大小选型一览表，每个扇区0.5K字节

第一扇区		第二扇区		第三扇区		第四扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
0000h	1FFh	200h	3FFh	400h	5FFh	600h	7FFh
第五扇区		第六扇区		第七扇区		第八扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
800h	9FFh	A00h	BFFh	C00h	DFFh	E00h	FFFh
第九扇区		第十扇区		第十一扇区		第十二扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
1000h	11FFh	1200h	13FFh	1400h	15FFh	1600h	17FFh
第十三扇区		第十四扇区		第十五扇区		第十六扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
1800h	19FFh	1A00h	1BFFh	1C00h	1DFFh	1E00h	1FFFh
第十七扇区		第十八扇区		第十九扇区		第二十扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
2000h	21FFh	2200h	23FFh	2400h	25FFh	2600h	27FFh
第二十一扇区		第二十二扇区		第二十三扇区		第二十四扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
2800h	29FFh	2A00h	2BFFh	2C00h	2DFFh	2E00h	2FFFH
第二十五扇区		第二十六扇区		第二十七扇区		第二十八扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
3000h	31FFh	3200h	33FFh	3400h	35FFh	3600h	37FFH
第二十九扇区		第三十扇区		第三十一扇区		第三十二扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
3800h	39FFh	3A00h	3BFFh	3C00h	3DFFh	3E00h	3FFFH
第三十三扇区		第三十四扇区		第三十五扇区		第三十六扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
4000h	41FFh	4200h	43FFh	4400h	45FFh	4600h	47FFH
第三十七扇区		第三十八扇区		第三十九扇区		第四十扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
4800h	49FFh	4A00h	4BFFh	4C00h	4DFFh	4E00h	4FFFH
第四十一扇区		第四十二扇区		第四十三扇区		第四十四扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
5000h	51FFh	5200h	53FFh	5400h	55FFh	5600h	57FFH
第四十五扇区		第四十六扇区		第四十七扇区		第四十八扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
5800h	59FFh	5A00h	5BFFh	5C00h	5DFFh	5E00h	5FFFH
第四十九扇区		第五十扇区		第五十一扇区		第五十二扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
6000h	61FFh	6200h	63FFh	6400h	65FFh	6600h	67FFH
第五十三扇区		第五十四扇区		第五十五扇区		第五十六扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
6800h	69FFh	6A00h	6BFFh	6C00h	6DFFh	6E00h	6FFFH
第五十七扇区		第五十八扇区		第五十九扇区		第六十扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
7000h	71FFh	7200h	73FFh	7400h	75FFh	7600h	77FFH
第六十一扇区		第六十二扇区		第六十三扇区		第六十四扇区	
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址
7800h	79FFh	7A00h	7BFFh	7C00h	7DFFh	7E00h	7FFFH

每个扇区512字节

建议一次修改的数据在同一扇区

这样跟通就普外部EEPROM用法一样

型号命名为IAP11xx或IAP10xx的单片机,没有独立的EEPROM,但是可利用IAP寄存器在程序区对程序Flash进行直接修改,做自己的在线编程/远程升级,当然,也可将程序Flash当EEPROM使用.扇区地址参照STC11/10xx系列单片机EEPROM详细地址表

举例如下:

IAP11F06,IAP11L06单片机的内部没有专门的EEPROM，但6K的程序区可以任意修改								
第一扇区		第二扇区		第三扇区		第四扇区		每个扇区512字节
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
0000h	1FFh	200h	3FFh	400h	5FFh	600h	7FFh	
第五扇区		第六扇区		第七扇区		第八扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
800h	9FFh	A00h	BFFh	C00h	DFFh	E00h	FFFh	
第九扇区		第十扇区		第十一扇区		第十二扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
1000h	11FFh	1200h	13FFh	1400h	15FFh	1600h	17FFh	

IAP10F14X,IAP10L14X单片机的内部没有专门的EEPROM，但14K的程序区可以任意修改								
第一扇区		第二扇区		第三扇区		第四扇区		每个扇区512字节
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
0000h	1FFh	200h	3FFh	400h	5FFh	600h	7FFh	
第五扇区		第六扇区		第七扇区		第八扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
800h	9FFh	A00h	BFFh	C00h	DFFh	E00h	FFFh	
第九扇区		第十扇区		第十一扇区		第十二扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
1000h	11FFh	1200h	13FFh	1400h	15FFh	1600h	17FFh	
第十三扇区		第十四扇区		第十五扇区		第十六扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
1800h	19FFh	1A00h	1BFFh	1C00h	1DFFh	1E00h	1FFFh	
第十七扇区		第十八扇区		第十九扇区		第二十扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
2000h	21FFh	2200h	23FFh	2400h	25FFh	2600h	27FFh	
第二十一扇区		第二十二扇区		第二十三扇区		第二十四扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
2800h	29FFh	2A00h	2BFFh	2C00h	2DFFh	2E00h	2FFFH	
第二十五扇区		第二十六扇区		第二十七扇区		第二十八扇区		
起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	起始地址	结束地址	
3000h	31FFh	3200h	33FFh	3400h	35FFh	3600h	37FFh	

4.3 IAP及EEPROM汇编简介

```
;用 DATA 还是 EQU 声明新增特殊功能寄存器地址要看你用的汇编器 / 编译器
    IAP_DATA        DATA    0C2h; 或    IAP_DATA        EQU     0C2h
    IAP_ADDRH       DATA    0C3h; 或    IAP_ADDRH       EQU     0C3h
    IAP_ADDRL       DATA    0C4h; 或    IAP_ADDRL       EQU     0C4h
    IAP_CMD         DATA    0C5h; 或    IAP_CMD         EQU     0C5h
    IAP_TRIG        DATA    0C6h; 或    IAP_TRIG        EQU     0C6h
    IAP_CONTR       DATA    0C7h; 或    IAP_CONTR       EQU     0C7h
;定义 ISP/IAP 命令及等待时间
    ISP_IAP_BYTE_READ       EQU 1       ;字节读
    ISP_IAP_BYTE_PROGRAM    EQU 2       ;字节编程,前提是该字节是空，0FFh
    ISP_IAP_SECTOR_ERASE    EQU 3       ;扇区擦除,要某字节为空，要擦一扇区
    WAIT_TIME       EQU 0               ;设置等待时间,30MHz 以下 0,24M 以下 1,
        ;20MHz 以下 2,12M 以下 3,6M 以下 4,3M 以下 5,2M 以下 6,1M 以下 7,
;字节读
    MOV   IAP_ADDRH, #BYTE_ADDR_HIGH        ;送地址高字节
    MOV   IAP_ADDRL, #BYTE_ADDR_LOW         ;送地址低字节
                                            ← 地址需要改变时才需重新送地址
    MOV   IAP_CONTR, #WAIT_TIME ;设置等待时间
    ORL   IAP_CONTR, #10000000B ;允许 ISP/IAP 操作
                                            ← 此两句可以合成一句,并且只送一次就够了
    MOV   IAP_CMD,          #ISP_IAP_BYTE_READ;送字节读命令,命令不需改变时,不需重新送命令
    MOV   IAP_TRIG,         #5Ah        ;先送 5Ah,再送 A5h 到 ISP/IAP 触发寄存器,每次都需如此
    MOV   IAP_TRIG,         #0A5h       ;送完 A5h 后，ISP/IAP 命令立即被触发起动
;CPU 等待 IAP 动作完成后，才会继续执行程序。
    NOP                                 ;数据读出到 IAP_DATA 寄存器后，CPU 继续执行程序
    MOV   A,        IAP_DATA            ;将读出的数据送往 Acc
;以下语句可不用,只是出于安全考虑而已
    MOV   IAP_CONTR, #00000000B         ;禁止 ISP/IAP 操作
    MOV   IAP_CMD,   #00000000B         ;去除 ISP/IAP 命令
    ;MOV  IAP_TRIG,  #00000000B         ;防止 ISP/IAP 命令误触发
    ;MOV  IAP_ADDRH, #80h                ;送地址高字节单元为 80h,指向非 EEPROM 区
    ;MOV  IAP_ADDRL, #00h                ;送地址低字节单元为 00h,防止误操作
```

```
;字节编程，该字节为 FFh/ 空时，可对其编程，否则不行,要先执行扇区擦除
    MOV   IAP_DATA,   #ONE_DATA     ;送字节编程数据到 IAP_DATA,只有数据改变时才需重新送
    MOV   IAP_ADDRH,  #BYTE_ADDR_HIGH  ;送地址高字节
    MOV   IAP_ADDRL,  #BYTE_ADDR_LOW   ;送地址低字节        地址需要改变时才需重新送地址
    MOV   IAP_CONTR,  #WAIT_TIME  ;设置等待时间
    ORL   IAP_CONTR,  #10000000B  ;允许 ISP/IAP 操作        此两句可合成一句,并且只送一次就够了
    MOV   IAP_CMD,    #ISP_IAP_BYTE_PROGRAM       ;送字节编程命令
    MOV   IAP_TRIG,   #5Ah            ;先送 5Ah,再送 A5h 到 ISP/IAP 触发寄存器,每次都需如此
    MOV   IAP_TRIG,   #0A5h           ;送完 A5h 后，ISP/IAP 命令立即被触发起动
;CPU 等待 IAP 动作完成后，才会继续执行程序.
    NOP                               ;字节编程成功后，CPU 继续执行程序
;以下语句可不用,只是出于安全考虑而已
    MOV   IAP_CONTR,  #00000000B  ;禁止 ISP/IAP 操作
    MOV   IAP_CMD,    #00000000B  ;去除 ISP/IAP 命令
    ;MOV  IAP_TRIG,   #00000000B  ;防止 ISP/IAP 命令误触发
    ;MOV  IAP_ADDRH,  #80h        ;送地址高字节单元为 80h,指向非 EEPROM 区,防止误操作
    ;MOV  IAP_ADDRL,  #00h        ;送地址低字节单元为 00h,指向非 EEPROM 区,防止误操作

;扇区擦除，没有字节擦除，只有扇区擦除，512 字节 / 扇区,每个扇区用得越少越方便
;如果要对某个扇区进行擦除，而其中有些字节的内容需要保留，则需将其先读到单片机
;内部的 RAM 中保存，再将该扇区擦除，然后将须保留的数据写回该扇区，所以每个扇区
;中用的字节数越少越好，操作起来越灵活越快.
;扇区中任意一个字节的地址都是该扇区的地址,无需求出首地址.

    MOV   IAP_ADDRH,  #SECTOR_FIRST_BYTE_ADDR_HIGH ;送扇区起始地址高字节   地址需要改变时
    MOV   IAP_ADDRL,  #SECTOR_FIRST_BYTE_ADDR_LOW  ;送扇区起始地址低字节   才需重新送地址
    MOV   IAP_CONTR,  #WAIT_TIME  ;设置等待时间
    ORL   IAP_CONTR,  #10000000B  ;允许 ISP/IAP      此两句可以合成一句,并且只送一次就够了
    MOV   IAP_CMD,    #ISP_IAP_SECTOR_ERASE;送扇区擦除命令,命令不需改变时,不需重新送命令
    MOV   IAP_TRIG,         #5Ah        ;先送 5Ah,再送 A5h 到 ISP/IAP 触发寄存器,每次都需如此
    MOV   IAP_TRIG,         #0A5h       ;送完 A5h 后，ISP/IAP 命令立即被触发起动
;CPU 等待 IAP 动作完成后，才会继续执行程序.
    NOP                                 ;扇区擦除成功后，CPU 继续执行程序
;以下语句可不用,只是出于安全考虑而已
    MOV   IAP_CONTR,  #00000000B       ;禁止 ISP/IAP 操作
    MOV   IAP_CMD,    #00000000B       ;去除 ISP/IAP 命令
    ;MOV  IAP_TRIG,   #00000000B       ;防止 ISP/IAP 命令误触发
    ;MOV  IAP_ADDRH,  #80h              ;送地址高字节单元为 80h,指向非 EEPROM 区
    ;MOV  IAP_ADDRL,  #00h              ;送地址低字节单元为 00h,防止误操作
```

小常识： (STC 单片机的 Data Flash 当 EEPROM 功能使用)

3 个基本命令 - - - - 字节读，字节编程，扇区擦除

字节编程：只能将“1”改为“0”，对“0”用字节编程是无用的。如果该字节是“1111,1111B”，则可将其中的“1”编程为“0”，如果该字节中有位为“0”，要将其改为“1”，则须先将整个扇区擦除，因为只有“扇区擦除”才可以将“0”变为“1”。

扇区擦除：只有“扇区擦除”才可能将“0”擦除为“1”。

大建议：

1.同一次修改的数据放在同一扇区中，不是同一次修改的数据放在另外的扇区,就不须读出保护。

2.如果一个扇区只用一个字节，那就是真正的 EEPROM,STC 单片机的 Data Flash 比外部 EEPROM 要快很多，读一个字节 / 编程一个字节大概是 0.2uS/60uS。

3.如果在一个扇区中存放了大量的数据，某次只需要修改其中的一个字节或一部分字节时，则另外的不需要修改的数据须先读出放在 STC 单片机的 RAM 中，然后擦除整个扇区，再将需要保留的数据和需修改的数据一并写回该扇区中。这时每个扇区使用的字节数是使用的越少越方便(不需读出一大堆需保留数据)。

4.4 一个完整的 EEPROM 测试程序，用宏晶的下载板可以直接测试

```
;STC11/10xx 系列单片机 EEPROM/IAP 功能测试程序演示
;/* ---  STC International Limited ------------------------------------------------ */
;/* ---  演示 STC11Fxx 系列 MCU EEPROM/IAP 功能 ---------------------------------- */
;本演示程序在 STC-ISP Ver 3.0A.PCB 的下载编程工具上测试通过,EEPROM 的数据在 P1 口上显示
;如果要在程序中使用或在文章中引用该程序,请在程序中或文章中注明使用了宏晶科技的资料及程序
;-----------------------------------------------
;声明与 IAP/ISP/EEPROM 有关的特殊功能寄存器的地址
IAP_DATA        EQU   0C2H
IAP_ADDRH       EQU   0C3H
IAP_ADDRL       EQU   0C4H
IAP_CMD         EQU   0C5H
IAP_TRIG        EQU   0C6H
IAP_CONTR       EQU   0C7H
;定义 ISP/IAP 命令
ISP_IAP_BYTE_READ       EQU   1H  ;字节读
ISP_IAP_BYTE_PROGRAM    EQU   2H  ;字节编程,可以将 1 写成 0，要将 1 变成 0，必须执行字节编程
ISP_IAP_SECTOR_ERASE    EQU   3H  ;扇区擦除,可以将 0 擦成 1，要将 0 变成 1，必须擦除整个扇区
;定义 Flash 操作等待时间及允许 IAP/ISP/EEPROM 操作的常数
;ENABLE_IAP      EQU   80H     ;系统工作时钟<30MHz 时，对 IAP_CONTR 寄存器设置此值
;ENABLE_IAP      EQU   81H     ;系统工作时钟<24MHz 时，对 IAP_CONTR 寄存器设置此值
 ENABLE_IAP      EQU   82H     ;系统工作时钟<20MHz 时，对 IAP_CONTR 寄存器设置此值
;ENABLE_IAP      EQU   83H     ;系统工作时钟<12MHz 时，对 IAP_CONTR 寄存器设置此值
;ENABLE_IAP      EQU   84H     ;系统工作时钟<6MHz 时，对 IAP_CONTR 寄存器设置此值
;ENABLE_IAP      EQU   85H     ;系统工作时钟<3MHz 时，对 IAP_CONTR 寄存器设置此值
;ENABLE_IAP      EQU   86H     ;系统工作时钟<2MHz 时，对 IAP_CONTR 寄存器设置此值
;ENABLE_IAP      EQU   87H     ;系统工作时钟<1MHz 时，对 IAP_CONTR 寄存器设置此值
DEBUG_DATA      EQU   5AH ;是本测试程序选定的 EEPROM 单元的数值如正确应等于的数值
;------------------------
;选择 MCU EEPROM 测试起始地址
DATA_FLASH_START_ADDRESS EQU 0000H  ;STC11/10xx 系列单片机的 EEPROM 测试起始地址
;-----------------------------------------------
    ORG   0000H
    LJMP  MAIN
;-----------------------------------------------
    ORG   0100H
MAIN:
    MOV     P1,#0F0H        ;演示程序开始工作,让 P1.0/P1.1/P1.2/P1.3 控制的灯亮
    LCALL    Delay          ;延时
    MOV     P1,#0FH         ;演示程序开始工作,让 P1.7/P1.6/P1.5/P1.4
```

```
    LCALL    Delay            ;延时
    MOV      SP,  #7FH        ;堆栈指针指向 7FH 单元
;*********************************************
;将 EEPROM 测试起始地址单元的内容读出
MAIN1:
    MOV     DPTR, #DATA_FLASH_START_ADDRESS   ;将 EEPROM 测试起始地址送 DPTR 数据指针
    LCALL   Byte_Read
    MOV     40H, A                  ;将 EEPROM 的值送 40H 单元保存
    CJNE    A, #DEBUG_DATA, DATA_NOT_EQU_DEBUG_DATA    ;如果数据比较不正确,就跳转

DATA_IS_DEBUG_DATA:
;数据是对的,亮 P1.7 控制的灯,然后在 P1 口上将 EEPROM 的数据显示出来
    MOV      P1,  #01111111B  ;如 (DATA_FLASH_START_ADDRESS)的值等于 #DEBUG_DATA, 亮 P1.7
    LCALL   Delay          ;延时
    MOV      A, 40H    ;将保存在 40H 单元中 EEPROM 的值从 40H 单元送累加器 A
    CPL     A          ;取反的目的是相应的灯亮代表 1,不亮代表 0
    MOV     P1,A      ;数据是对的，送 P1 显示
WAIT1:
    SJMP    WAIT1    ;数据是对的，送 P1 显示后，CPU 在此无限循环执行此句

DATA_NOT_EQU_DEBUG_DATA:
;EEPROM 里的数据是错的,亮 P1.3 控制的灯,然后在 P1 口上将错误的数据显示出来,
;再将该 EEPROM 所在的扇区整个擦除,将正确的数据写入后,亮 P1.5 控制的灯
    MOV     P1,  #11110111B ;如 (DATA_FLASH_START_ADDRESS)的值不等于 #DEBUG_DATA, 亮 P1.3
    LCALL   Delay        ;延时
    MOV      A,    40H   ;将保存在 40H 单元中 EEPROM 的值从 40H 单元送累加器 A
    CPL     A           ;取反的目的是相应的灯亮代表 1,不亮代表 0
    MOV     P1,  A     ;数据不对，送 P1 显示
    LCALL   Delay;延时

    MOV     DPTR,#DATA_FLASH_START_ADDRESS   ;将 EEPROM 测试起始地址送 DPTR 数据指针
    ACALL   Sector_Erase   ;擦除整个扇区
    MOV     DPTR, #DATA_FLASH_START_ADDRESS   ;将 EEPROM 测试起始地址送 DPTR 数据指针
    MOV     A,     #DEBUG_DATA        ;写入 EEPROM 的数据为 #DEBUG_DATA
    ACALL   Byte_Program              ;字节编程
    MOV     P1,    #11011111B         ;将先前亮的 P1.3 灯关闭 ,再亮 P1.5 灯,代表数据已被修改
WAIT2:
    SJMP    WAIT2         ;字节编程后,CPU 在此无限循环执行此句
;*********************************************
```

```
;-------------------------------------------
;读一字节，调用前需打开 IAP 功能，入口:DPTR = 字节地址，返回:A = 读出字节
Byte_Read:
    MOV     IAP_CONTR,  #ENABLE_ISP     ;打开 IAP 功能，设置 Flash 操作等待时间
    MOV     IAP_CMD,    #ISP_IAP_BYTE_READ  ;设置为 IAP/ISP/EEPROM 字节读模式命令
    MOV     IAP_ADDRH,  DPH             ;设置目标单元地址的高 8 位地址
    MOV     IAP_ADDRL,  DPL             ;设置目标单元地址的低 8 位地址
    ;CLR     EA
    MOV     IAP_TRIG,   #5AH        ;先送 5Ah,再送 A5h 到 ISP/IAP 触发寄存器,每次都需如此
    MOV     IAP_TRIG,   #0A5H       ;送完 A5h 后，ISP/IAP 命令立即被触发起动
    NOP
    MOV     A,    IAP_DATA          ;读出的数据在 IAP_DATA 单元中,送入累加器 A
    ;SETB    EA
    ACALL IAP_Disable ;关闭 IAP 功能，清相关的特殊功能寄存器,使 CPU 处于安全状态,
                      ;一次连续的 IAP 操作完成之后建议关闭 IAP 功能,不需要每次都关
    RET
;-------------------------------------------
;字节编程，调用前需打开 IAP 功能，入口:DPTR = 字节地址，A= 须编程字节的数据
Byte_Program:
    MOV   IAP_CONTR, #ENABLE_ISP    ;打开 IAP 功能，设置 Flash 操作等待时间
    MOV   IAP_CMD, #ISP_IAP_BYTE_PROGRAM    ;设置为 IAP/ISP/EEPROM 字节编程模式命令
    MOV   IAP_ADDRH, DPH            ;设置目标单元地址的高 8 位地址
    MOV   IAP_ADDRL, DPL            ;设置目标单元地址的低 8 位地址
    MOV   IAP_DATA, A               ;要编程的数据先送进 IAP_DATA 寄存器
    ;CLR   EA
    MOV   IAP_TRIG, #5AH            ;先送 5Ah,再送 A5h 到 ISP/IAP 触发寄存器,每次都需如此
    MOV   IAP_TRIG, #0A5H           ;送完 A5h 后，ISP/IAP 命令立即被触发起动
    NOP
    ;SETB  EA
     ACALL IAP_Disable ;关闭 IAP 功能，清相关的特殊功能寄存器,使 CPU 处于安全状态,
                       ;一次连续的 IAP 操作完成之后建议关闭 IAP 功能,不需要每次都关
    RET
;-------------------------------------------
;擦除扇区，入口:DPTR = 扇区地址
Sector_Erase:
    MOV   IAP_CONTR, #ENABLE_ISP    ;打开 IAP 功能，设置 Flash 操作等待时间
    MOV   IAP_CMD, #03H             ;设置为 IAP/ISP/EEPROM 扇区擦除模式命令
    MOV   IAP_ADDRH, DPH            ;设置目标单元地址的高 8 位地址
    MOV   IAP_ADDRL, DPL            ;设置目标单元地址的低 8 位地址
    ;CLR   EA
    MOV   IAP_TRIG, #5AH            ;先送 5Ah,再送 A5h 到 ISP/IAP 触发寄存器,每次都需如此
    MOV   IAP_TRIG, #0A5H           ;送完 A5h 后，ISP/IAP 命令立即被触发起动
    NOP
    ;SETB  EA
     ACALL IAP_Disable ;关闭 IAP 功能，清相关的特殊功能寄存器,使 CPU 处于安全状态,
                       ;一次连续的 IAP 操作完成之后建议关闭 IAP 功能,不需要每次都关
    RET
```

```
;-----------------------------------------------
IAP_Disable:
;关闭 IAP 功能，清相关的特殊功能寄存器,使 CPU 处于安全状态，
;一次连续的 IAP 操作完成之后建议关闭 IAP 功能,不需要每次都关
    MOV   IAP_CONTR, #0             ;关闭 IAP 功能
    MOV   IAP_CMD,  #0              ;清命令寄存器,使命令寄存器无命令,此句可不用
    MOV   IAP_TRIG, #0              ;清命令触发寄存器,使命令触发寄存器无触发,此句可不用
    MOV   IAP_ADDRH,  #80h              ;送地址高字节单元为 80h,指向非 EEPROM 区
    MOV   IAP_ADDRL,  #00h              ;送地址低字节单元为 00h,防止误操作
    RET
;-----------------------------------------------
Delay:
    CLR   A
    MOV   R0, A
    MOV   R1, A
    MOV   R2, #20H
Delay_Loop:
    DJNZ  R0, Delay_Loop
    DJNZ  R1, Delay_Loop
    DJNZ  R2, Delay_Loop
    RET

;-----------------------------------------------

    END
;***********************************************
```

第五章　STC11/10xx 系列单片机定时器应用

5.1　定时器0/1的介绍

STC11/10xx 系列有 2 个定时器，定时器 0 和定时器 1 两个 16 位定时器，与传统 8051 的定时器完全兼容，也可以设置为 1T 模式，其中在定时器 1 做波特率发生器时，定时器 0 可以当两个 8 位定时器用。

定时器 0 和 1

定时和计数功能由特殊功能寄存器 TMOD 的控制位 C/$\overline{T}$ 进行选择，TMOD 寄存器的各位信息如下表所列。可以看出，2 个定时 / 计数器有 4 种操作模式，通过 TMOD 的 M1 和 M0 选择。2 个定时 / 计数器的模式 0、1 和 2 都相同，模式 3 不同，各模式下的功能如下所述。

寄存器TMOD各位的功能描述

TMOD　　地址：89H　　复位值：00H

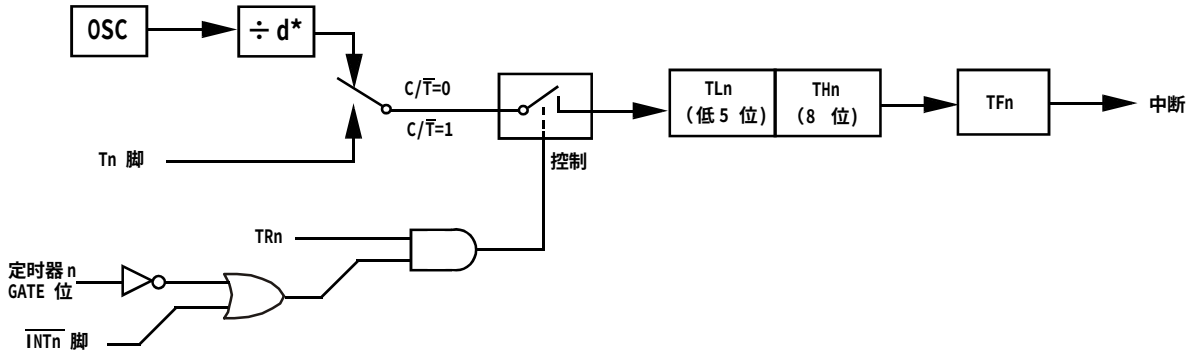
不可位寻址

7	6	5	4	3	2	1	0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
定时器 1				定时器 0			

位	符号	功能
TMOD.7/	GATE	TMOD.7 控制定时器 1,置 1 时只有在 $\overline{INT1}$ 脚为高及 TR1 控制位置 1 时才可打开定时器 / 计数器 1。
TMOD.3/	GATE	TMOD.3 控制定时器 0,置 1 时只有在 $\overline{INT0}$ 脚为高及 TR0 控制位置 1 时才可打开定时器 / 计数器 0。
TMOD.6/	C/$\overline{T}$	TMOD.6 控制定时器 1 用作定时器或计数器，清零则用作定时器（从内部系统时钟输入），置 1 用作计数器（从 T1/P3.5 脚输入）
TMOD.2/	C/$\overline{T}$	TMOD.2 控制定时器 0 用作定时器或计数器，清零则用作定时器（从内部系统时钟输入），置 1 用作计数器（从 T0/P3.4 脚输入）
TMOD.5/TMOD.4	M1、M0	定时器定时器 / 计数器 1 模式选择
	0　0	13 位定时器 / 计数器，兼容 8048 定时器模式，TL1 只用低 5 位参与分频，TH1 整个 8 位全用。
	0　1	16 位定时器 / 计数器，TL1、TH1 全用
	1　0	8 位自动重装载定时器，当溢出时将 TH1 存放的值自动重装入 TL1。
	1　1	定时器 / 计数器 1 此时无效（停止计数）。
TMOD.1/TMOD.0	M1、M0	定时器 / 计数器 0 模式选择
	0　0	13 位定时器 / 计数器，兼容 8048 定时器模式，TL0 只用低 5 位参与分频，TH0 整个 8 位全用。
	0　1	16 位定时器 / 计数器，TL0、TH0 全用
	1　0	8 位自动重装载定时器，当溢出时将 TH0 存放的值自动重装入 TL0。
	1　1	定时器 0 此时作为双 8 位定时器 / 计数器。TL0 作为一个 8 位定时器 / 计数器，通过标准定时器 0 的控制位控制。TH0 仅作为一个 8 位定时器,由定时器 1 的控制位控制。

1.模式 0

将定时器设置成模式 0 时类似 8048 定时器，即 8 位计数器带 32 分频的预分频器。下图所示为模式 0 工作方式。此模式下，定时器配置为 13 位的计数器，由 TLn 的低 5 位和 THn 的 8 位所构成。TLn 低 5 位溢出向 THn 进位，THn 计数溢出置位 TCON 中的溢出标志位 TFn（n=0，1）。GATE=0 时，如 TRn=1，则定时器计数。GATE=1 时，允许由外部输入 $\overline{INT1}$ 控制定时器 1，INT0 控制定时器 0，这样可实现脉宽测量。TRn 为 TCON 寄存器内的控制位，TCON 寄存器各位的具体功能描述见 TCON 寄存器各位的具体功能描述表。



* 在 T0x12 = 0 模式下，d=12(12 时钟模式)； 在 T0x12 = 1 模式下，d=1(1T)。

图 定时器 / 计数器 0 和定时器 / 计数器 1 的模式 0 ： 13 位定时 / 计数器

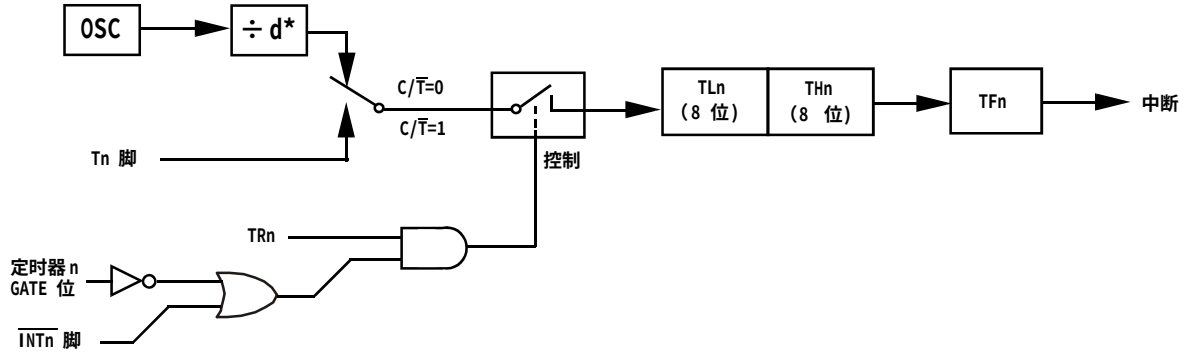
表 寄存器 TCON 各位的功能描述

TCON 地址：88H										
可位寻址 复位值：00H		7	6	5	4	3	2	1	0	
		TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
位	符 号	功 能								
TCON.7	TF1	定时器 / 计数器 1 溢出标志位。当 T1 被允许计数后，T1 从初值开始加 1 计数，最高位产生溢出时，置“ 1 ”TF1，并向 CPU 请求中断，当 CPU 响应时，由硬件清“ 0 ”TF1，TF1 也可以由程序查询或清“ 0 ”。								
TCON.6	TR1	定时器 T1 的运行控制位。该位由软件置位和清零。当 GATE（TMOD.7）=0，TR1=1 时就允许 T1 开始计数，TR1=0 时禁止 T1 计数。当 GATE（TMOD.7）=1，TR1=1 且 INT1 输入高电平时，才允许 T1 计数。								
TCON.5	TF0	定时器 / 计数器 0 溢出标志位。当 T0 被允许计数后，T0 从初值开始加 1 计数，最高位产生溢出时，置“ 1 ”TF0，并向 CPU 请求中断，当 CPU 响应时，由硬件清“ 0 ”TF0，TF0 也可以由程序查询或清“ 0 ”。								
TCON.4	TR0	定时器 T0 的运行控制位。该位由软件置位和清零。当 GATE（TMOD.3）=0，TR0=1 时就允许 T0 开始计数，TR1=0 时禁止 T0 计数。当 GATE（TMOD.3）=1，TR0=1 且 $\overline{\text{INT0}}$ 输入高电平时，才允许 T0 计数。								
TCON.3	IE1	外部中断 1 中断请求标志位。当主机响应中断转向该中断服务程序执行时，由内部硬件自动将 IE1 位清 0。								
TCON.2	IT1	外部中断 1 触发方式控制位。IT1=0 时，外部中断 1 为低电平触发方式，当 $\overline{\text{INT1}}$（P3.3）输入低电平时，置位 IE1。采用低电平触发方式时，外部中断源（输入到 INT1）必须保持低电平有效，直到该中断被 CPU 响应，同时在该中断服务程序执行完之前，外部中断源必须被清除（P3.3 要变高），否则将产生另一次中断。当 IT1=1 时，则外部中断 1（$\overline{\text{INT1}}$）端口由“ 1 ”→“ 0 ”下降沿跳变，激活中断请求标志位 IE1，向主机请求中断处理。								
TCON.1	IE0	外部中断 0 中断请求标志位。当主机响应中断转向该中断服务程序执行时，由内部硬件自动将 IE0 位清 0。								
TCON.0	IT0	外部中断 0 触发方式控制位。IT0=0 时，外部中断 0 为低电平触发方式，当 $\overline{\text{INT0}}$（P3.2）输入低电平时，置位 IE0。采用低电平触发方式时，外部中断源（输入到 $\overline{\text{INT0}}$）必须保持低电平有效，直到该中断被 CPU 响应，同时在该中断服务程序执行完之前，外部中断源必须被清除（P3.2 要变高），否则将产生另一次中断。当 IT0=1 时，则外部中断 0（$\overline{\text{INT0}}$）端口由“ 1 ”→“ 0 ”下降沿跳变，激活中断请求标志位 IE1，向主机请求中断处理。								

该 13 位寄存器包含 THn 全部 8 个位及 TLn 的低 5 位。TLn 的高 3 位不定，可将其忽略。置位运行标志（TRn）不能清零此寄存器。模式 0 的操作对于定时器 0 及定时器 1 都是相同的。2 个不同的 GATE 位（TMOD.7 和 TMOD.3）分别分配给定时器 1 及定时器 0。

2.模式 1

模式 1 除了使用了 THn 及 TLn 全部 16 位外，其他与模式 0 完全相同。

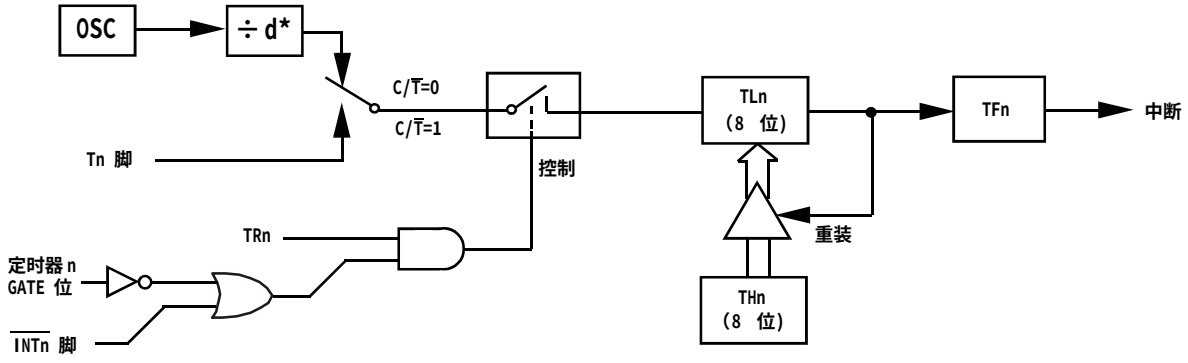


* 在 T0x12 = 0 模式下，d=12(12 时钟模式)； 在 T0x12 = 1 模式下，d=1(1T)。

图 定时器 / 计数器 0 和定时器 / 计数器 1 的模式 1 ： 16 位定时 / 计数器

3.模式 2

此模式下定时器 / 计数器 0 和 1 作为可自动重装载的 8 位计数器（TLn），如下图所示。TLn 的溢出不仅置位 TFn，而且将 THn 内容重新装入 TLn，THn 内容由软件预置，重装时 THn 内容不变。模式 2 的操作对于定时器 0 及定时器 1 是相同的。



* 在 T0x12 = 0 模式下，d=12(12 时钟模式)； 在 T0x12 = 1 模式下，d=1(1T)。

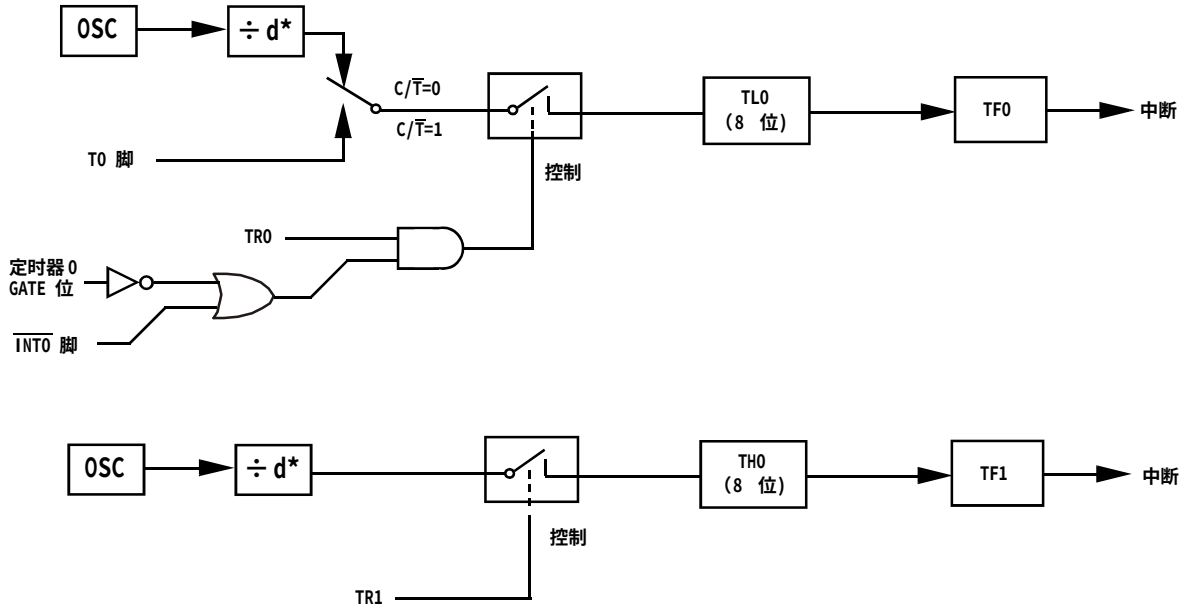
图 定时器 / 计数器 0 和 1 的模式 2 ： 8 位自动重装载

4.模式 3

对定时器 1，在模式 3 时，定时器 1 停止计数，效果与将 TR1 设置为 0 相同。

对定时器 0，此模式下定时器 0 的 TL0 及 TH0 作为 2 个独立的 8 位计数器。下图为模式 3 时的定时器 0 逻辑图。TL0 占用定时器 0 的控制位：C/$\overline{T}$、GATE、TR0、$\overline{INT0}$ 及 TF0。TH0 限定为定时器功能（计数器周期），占用定时器 1 的 TR1 及 TF1。此时，TH0 控制定时器 1 中断。

模式 3 是为了增加一个附加的 8 位定时器 / 计数器而提供的，使单片机具有三个定时器 / 计数器。模式 3 只适用于定时器 / 计数器 0，定时器 T1 处于模式 3 时相当于 TR1=0，停止计数（此时 T1 可用来作串行口波特率发生器），而 T0 可作为两个定时器用。



* 在 T0x12 = 0 模式下，d=12(12 时钟模式)； 在 T0x12 = 1 模式下，d=1(1T)。

图 定时 / 计数器 0 的模式 3 ： 两个 8 位计数器

5.也可将定时器 0 和定时器 1 设置为 1T 模式

STC11/10xx 系列是 1T 的 8051 单片机，为兼容传统 8051，定时器 0 和定时器 1 复位后是传统 8051 的速度，即 12 分频，这是为了兼容传统 8051。但也可不进行 12 分频，通过设置新增加的特殊功能寄存器 AUXR，将 T0,T1 设置为 1T。普通 111 条机器指令是固定的，快 3 到 24 倍，无法改变。

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	-	-	-	-	-	0000,000

T0x12： 0，定时器 0 是传统 8051 速度，12 分频；1，定时器 0 的速度是传统 8051 的 12 倍，不分频

T1x12： 0，定时器 1 是传统 8051 速度，12 分频；1，定时器 1 的速度是传统 8051 的 12 倍，不分频

如果 UART 串口用定时器 1 做波特率发生器，T1x12 位就可以控制 UART 串口是 12T 还是 1T 了。

UART 串口的模式 0：

STC11/10xx 系列是 1T 的 8051 单片机，为了兼容传统 8051，UART 串口复位后是兼容传统 8051 的。

UART_M0x6： 0，UART 串口的模式 0 是传统 12T 的 8051 速度，12 分频；

1，UART 串口的模式 0 的速度是传统 12T 的 8051 的 6 倍，2 分频

如果用定时器 T1 做波特率发生器时,UART 串口的速度由 T1 的溢出率决定

5.2　定时器0/1应用举例

【例1】　定时/计数器编程，定时/计数器的应用编程主要需考虑：根据应用要求，通过程序初始化，正确设置控制字，正确计算和计算计数初值，编写中断服务程序，适时设置控制位等。通常情况下，设置顺序大致如下：

1)工作方式控制字（TMOD、T2CON）的设置；

2)计数初值的计算并装入THx、TLx、RCAP2H、RCAP2L；

3)中断允许位ETx、EA的设置，使主机开放中断；

4)　启/停位TRx的设置等。

现以定时/计数器0或1为例作一简要介绍。

8051系列单片机的定时器/计数器0或1是以不断加1进行计数的，即属加1计数器，因此，就不能直接将实际的计数值作为计数初值送入计数寄存器THx、TLx中去，而必须将实际计数值以2^8、2^{13}、2^{16}为模求补，以其补码作为计数初值设置THx和TLx。

设：实际计数值为X，计数器长度为n（n=8、13、16），则应装入计数器THx、TLx中的计数初值为2^n-x，式中2^n为取模值。例如，工作方式0的计数长度为13位，则n=13，以2^{13}为模，工作方式1的计数长度为16，则n=16，以2^{16}为模等等。所以，计数初值为（x）=　2^n-x。

对于定时模式，是对机器周期计数，而机器周期与选定的主频密切相关。因此，需根据应用系统所选定的主频计算出机器周期值。现以主频6MHz为例，则机器周期为：

$$\text{一个机器周期}=\frac{12}{\text{主振频率}}=\frac{12}{6\times10^6}\ \mu s\ =2\ \mu s$$

$$\text{实际定时时间}\ Tc\ =\ x\cdot Tp$$

式中Tp为机器周期，Tc为所需定时时间，x为所需计数次数。Tp和Tp一般为已知值，在求出Tp后即可求得所需计数值x，再将x求补码，即求得定时计数初值。即

$$(x)\text{补}=2^n-x$$

例如，设定时时间Tc　=　5ms，机器周期TP　=　2 μs，可求得定时计数次数

$$x=\frac{5ms}{2\ \mu s}=2500\text{次}$$

设选用工作方式1，则n=16，则应设置的定时时间计数初值为：(x)补=　2^{16}　-　x=65536-2500=63036，还需将它分解成两个8位十六进制数，分别求得低8位为3CH装入TLx，高8位为F6H装入THx中。

工作方式0、1、2的最大计数次数分别为8192、65536和256。

对外部事件计数模式，只需根据实际计数次数求补后变换成两个十六进制码即可。

【例2】　定时/计数器应用编程，设某应用系统，选择定时/计数器1定时模式，定时时间Tc　=　10ms，主频频率为12MHz，每10ms向主机请求处理。选定工作方式1。计算得计数初值：低8位初值为F0H，高8位初值为D8H。

（1）初始化程序

所谓初始化，一般在主程序中根据应用要求对定时 / 计数器进行功能选择及参数设定等预置程序，本例初始化程序如下：

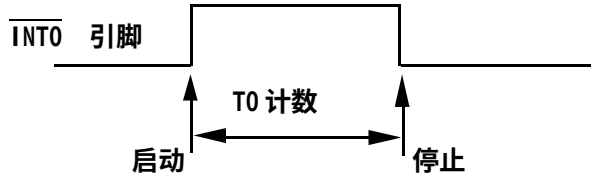
```
START：
        ⋮                          ；主程序段
        MOV   SP，#60H             ；设置堆栈区域
        MOV   TMOD，#10H       ；选择 T1 、定时模式，工作方式 1
        MOV   TH1，#0D8H       ；设置高字节计数初值
        MOV   TL1，#0F0H       ；设置低字节计数初值
        SETB  EA                   } 开中断
        SETB  ET1                  }
        ⋮                          ；其他初始化程序
        SETB  TR1                  ；启动 T1 开始计时
        ⋮                          ；继续主程序
```

（2 ）中断服务程序

```
INTT1：  PUSH  A                   }
         PUSH DPL                  } 现场保护
         PUSH DPH                  }
          ⋮
         MOV   TL1,#0F0H           } 重新置初值
         MOV   TH1,#0D8H           }
          ⋮                        ；中断处理主体程序
         POP   DPH                 }
         POP   DPL                 } 现场恢复
         POP   A                   }
         RETI                      ；返回
```

这里展示了中断服务子程序的基本格式。8052 系列单片机的中断属于矢量中断，每一个矢量中断源只留有 8 个字节单元，一般是不够用的，常需用转移指令转到真正的中断服务子程序区去执行。

【例 3】 对外部正脉冲测宽。选择定时 / 计数器 2 进行脉宽测试较方便，但也可选用定时 / 计数器 0 或定时 / 计数器 1 进行测宽操作。本例选用定时 / 计数器 0（T0）以定时模式，工作方式 1 对 $\overline{\text{INT0}}$ 引脚上的正脉冲进行脉宽测试。



设置 GATE 为 1，机器周期 TP 为 1 μ s。本例程序段编制如下：

```
INTT0：  MOV   TMOD，#09H      ；设 T0 为定时方式 1，GATE 为 1
```

```
            MOV   TL0，#00H          ；
            MOV   TH0，#00H          ；} TH0,TL0 清 0
            CLR   EX0                ；关 INT0 中断
LOP1：      JB    P3.2，LOP1     ；等待 INT0 引低电平
LOP2：      JNB   P3.2，LOP2     ；等待 INT0 引脚高电平
            SETB  TR0                ；启动 T0 开始计数
LOP3：      JB    P3.2，LOP3     ；等待 INT0 低电平
            CLR   TR0                ；停止 T0 计数
            MOV   A，TL0             ；低字节计数值送 A
            MOV   B，TH0             ；高字节计数值送 B
             ⋮                       ；计算脉宽和处理
```

【例 4】 利用定时 / 计数器 0 或定时 / 计数器 1 的 Tx 端口改造成外部中断源输入端口的应用设计。

在某些应用系统中常会出现原有的两个外部中断源 INT0 和 INT1 不够用，而定时 / 计数器有多余，则可将 Tx 用于增加的外部中断源。现选择定时 / 计数器 1 为对外部事件计数模式工作方式 2（自动再装入），设置计数初值为 FFH，则 T1 端口输入一个负跳变脉冲，计数器即回 0 溢出，置位对应的中断请求标志位 TF1 为 1，向主机请求中断处理，从而达到了增加一个外部中断源的目的。应用定时 / 计数器 1（T1）的中断矢量转入中断服务程序处理。其程序示例如下：

（1 ）主程序段：

```
     ORG   0000H
     AJMP  MAIN               ；转主程序
     ORG   001BH
     LJMP  INTER              ；转 T1 中断服务程序
      ⋮
     ORG   0100               ；主程序入口
MAIN：…
      ⋮
     MOV   SP，#60H           ；设置堆栈区
     MOV   TMOD，#60H ；设置定时 / 计数器 1，计数方式 2
     MOV   TL1，#0FFH ；设置计数常数
     MOV   TH1，#0FFH
     SETB  EA                 ；开中断
     SETB  ET1                ；开定时 / 计数器 1 中断
     SETB  TR1                ；启动定时 / 计数器 1 计数
      ⋮
```

（2 ）中断服务程序（具体处理程序略）

```
            ORG   1000H
INTER：     PUSH  A                  ；
            PUSH  DPL                ；} 现场入栈保护
            PUSH  DPH                ；
             ⋮
```

```
        ⋮                          } 中断处理主体程序
        ⋮
      POP   DPH                    }
      POP   DPL                    } 现场出栈复原
      POP   A                      }
      RETI                         ；返回
```

这是中断服务程序的基本格式。

【例 5】 某应用系统需通过 P1.0 和 P1.1 分别输出周期为 200 μs 和 400 μs 的方波。为此，系统选用定时器 / 计数器 0（T0），定时方式 3，主频为 6MHz，TP=2 μs，经计算得定时常数为 9CH 和 38H。

本例程序段编制如下：

（1）初始化程序段

```
        ⋮
PLT0:MOV   TMOD,#03H          ；设置 T0 定时方式 3
     MOV   TL0，#9CH          ；设置 TL0 初值
     MOV   TH0，#38H          ；设置 TH0 初值
     SETB  EA                 ；
     SETB  ET0                ；} 开中断
     SETB  ET1                ；
     SETB  TR0                ；启动
     SETB  TR1                ；启动
        ⋮
```

(2)中断服务程序段

1)

```
INT0P：  ⋮
         ⋮
     MOV   TL0，#9CH          ；重新设置初值
     CPL   P1.0               ；对 P1.0 输出信号取反
         ⋮
     RETI                     ；返回
```

2)

```
INT1P    ⋮
         ⋮
     MOV   TH0，#38H          ；重新设置初值
     CPL   P1.1               ；对 P1.1 输出信号取反
         ⋮
     RETI                     ；返回
```

在实际应用中应注意的问题如下。

（1）定时 / 计数器的实时性

定时 / 计数器启动计数后，当计满回 0 溢出向主机请求中断处理，由内部硬件自动进行。但从回 0 溢出请求中断到主机响应中断并作出处理存在时间延迟，且这种延时随中断请求时的现场环境的不同而不同，一般需延时 3 个机器周期以上，这就给实时处理带来误差。大多数应用场合可忽略不计，但对某些要求实时性苛刻的场合，应采用补偿措施。

这种由中断响应引起的时间延时，对定时 / 计数器工作于方式 0 或 1 而言有两种含义：一是由于中断响应延时而引起的实时处理的误差；二是如需多次且连续不间断地定时 / 计数，由于中断响应延时，则在中断服务程序中再置计数初值时已延误了若干个计数值而引起误差，特别是用于定时就更明显。

例如选用定时方式 1 设置系统时钟，由于上述原因就会产生实时误差。这种场合应采用动态补偿办法以减少系统始终误差。所谓动态补偿，即在中断服务程序中对 THx、TLx 重新置计数初值时，应将 THx、TLx 从回 0 溢出又重新从 0 开始继续计数的值读出，并补偿到原计数初值中去进行重新设置。可考虑如下补偿方法：

```
    ⋮
CLR   EA              ；禁止中断
MOV   A，TLx          ；读 TLx 中已计数值
ADD   A，#LOW         ；LOW 为原低字节计数初值
MOV   TLx，A          ；设置低字节计数初值
MOV   A，#HIGH        ；原高字节计数初值送 A
ADDC  A，THx          ；高字节计数初值补偿
MOV   THx，A          ；置高字节计数初值
SETB  EA              ；开中断
    ⋮
```

（2）动态读取运行中的计数值

在动态读取运行中的定时 / 计数器的计数值时，如果不加注意，就可能出错。这是因为不可能在同一时刻同时读取 THx 和 TLx 中的计数值。比如，先读 TLx 后读 THx，因为定时 / 计数器处于运行状态，在读 TLx 时尚未产生向 THx 进位，而在读 THx 前已产生进位，这时读得的 THx 就不对了；同样，先读 THx 后读 TLx 也可能出错。

一种可避免读错的方法是：先读 THx，后读 TLx，将两次读得的 THx 进行比较；若两次读得的值相等，则可确定读的值是正确的，否则重复上述过程，重复读得的值一般不会再错。此法的软件编程如下：

```
RDTM：  MOV  A，THx                ；读取 THx 存 A 中
        MOV  R0，TLx               ；读取 TLx 存 R0 中
        CJNE A，THx，RDTM          ；比较两次 THx 值，若相等，则读得的值正
                                   ；确，程序往下执行，否则重读
        MOV  R1，A                 ；将 THx 存于 R1 中
          ⋮
```

5.3 用定时器1做波特率发生器

```
;/* ---  STC International Limited ---------------------------------------------- */
;/* ---  宏晶科技 姚永平  设计 2006/1/6   V1.0 ---------------------------------------- */
;/* ---  演示 STC11/10xx 系列 MCU 定时器1作波特率发生器功能 ------------------------- */
;本演示程序在宏晶的STC-ISP Ver 3.0A.PCB的下载编程工具上测试通过
;如果要在程序中使用或在文章中引用该程序,请在程序中或文章中注明使用了宏晶科技的资料及程序
;-------------------------------------------------------------------------------
;     本程序演示STC11/10xx 系列单片机用定时器1作 RS-232 通信
;波特率发生器的使用方法，有关波特率自动重装数的计算请查看程序后面的内容
;     STC11/10xx 系列是 "一个时钟/机器周期" 的 8051单片机。它
;的定时器0、定时器1  有两种计数速率，一种是12T模式：每12个时钟加 1，与普通的
;8051单片机相同；另一种是1T模式：每个时钟加 1，是普通8051单片机的 12 倍。
;     STC89C51RC/RD+ 系列是 "12个时钟/机器周期" 的 8051单片机，与普通的 8051单片
;机相同。
;     STC11/10xx 系列的单片机，定时器0、定时器1 的计数速率由
;特殊功能寄存器 AUXR 的 bit7，bit6 决定，bit7 的符号是 T0x12，如果 T0x12=1，
;定时器0 就工作在 1T模式。bit6 的符号是 T1x12，如果 T1x12=1，定时器1 就工作在
;1T模式。有关详情请参考 STC11/10xx 系列单片机器件手册(中文应用指南)。

;使用方法:
;  1．修改程序，改变波特率参数或改变定时器1的计数速率(1T模式/12T模式)
;  2．汇编程序，将代码下载到单片机中
;  3．调整串口调试助手的波特率与单片机的波特率相同，并打开调试助手的串口。STC
;     下载程序 STC-ISP.exe 版本 3.2 以上有串口调试助手功能。
;  4．打开单片机电源，可以在串口调试助手的接收区看到单片机发出的数据
;  5．用串口调试助手发送单个字节到单片机，单片机收到后会立即回发到串口调试助手
;  6．反复步骤 1-5，检验波特率参数是否正确，特别要观察定时器1 工作在 1T模式
;     的波特率。例如，先设置定时器1 工作在 12T模式，设置波特率为 9600，执行
;     步骤 2-5，检验波特率参数是否正确。然后仅仅将定时器1 的计数速率改成
;     1T模式，执行步骤 2-5，就会发现本程序的波特率变成了 115200，波特率是
;     12T模式的 12 倍。
;
```

```
;定义 STC11/10xx 系列 MCU 特殊功能寄存器
AUXR    EQU  8EH
;-------------------------------------------------------------------------
;定义波特率自动重装数
;*************************************************************************
;以下是 Fosc = 22.1184MHz，1T 模式，SMOD=1 时，计算出的自动重装数和波特率
;RELOAD_COUNT EQU 0FFH      ;Baud=1,382,400 bps
;RELOAD_COUNT EQU 0FEH      ;Baud=691,200 bps
;RELOAD_COUNT EQU 0FDH      ;Baud=460,800 bps
;RELOAD_COUNT EQU 0FCH      ;Baud=345,600 bps
;RELOAD_COUNT EQU 0FBH      ;Baud=276,480 bps
;RELOAD_COUNT EQU 0FAH      ;Baud=230,400 bps
;RELOAD_COUNT EQU 0F4H      ;Baud=115,200 bps
;RELOAD_COUNT EQU 0E8H      ;Baud=57,600 bps
;RELOAD_COUNT EQU 0DCH      ;Baud=38,400 bps
;RELOAD_COUNT EQU 0B8H      ;Baud=19,200 bps
;RELOAD_COUNT EQU 70H      ;Baud=9,600 bps
;以上是 Fosc = 22.1184MHz，1T 模式，SMOD=1 时，计算出的自动重装数和波特率
;*************************************************************************

;*************************************************************************
;以下是 Fosc = 1.8432MHz，1T 模式，SMOD=1 时，计算出的自动重装数和波特率
;RELOAD_COUNT EQU 0FFH      ;Baud=115,200 bps
;RELOAD_COUNT EQU 0FEH      ;Baud=57,600 bps
;RELOAD_COUNT EQU 0FDH      ;Baud=38,400 bps
;RELOAD_COUNT EQU 0FCH      ;Baud=28,800 bps
;RELOAD_COUNT EQU 0FAH      ;Baud=19,200 bps
;RELOAD_COUNT EQU 0F4H      ;Baud=9,600 bps
;RELOAD_COUNT EQU 0E8H      ;Baud=4,800 bps
;RELOAD_COUNT EQU 0D0H      ;Baud=2,400 bps
;RELOAD_COUNT EQU 0A0H      ;Baud=1,200 bps
;以上是 Fosc = 1.8432MHz，1T 模式，SMOD=1 时，计算出的自动重装数和波特率
;*************************************************************************

;*************************************************************************
;以下是 Fosc = 18.432MHz，1T 模式，SMOD=1 时，计算出的自动重装数和波特率
;RELOAD_COUNT EQU 0FFH      ;Baud=1,152,000 bps
;RELOAD_COUNT EQU 0FEH      ;Baud=576,000 bps
;RELOAD_COUNT EQU 0FDH      ;Baud=288,000 bps
;RELOAD_COUNT EQU 0FCH      ;Baud=144,000 bps
;RELOAD_COUNT EQU 0F6H      ;Baud=115,200 bps
;RELOAD_COUNT EQU 0ECH      ;Baud=57,600 bps
;RELOAD_COUNT EQU 0E2H      ;Baud=38,400 bps
;RELOAD_COUNT EQU 0D8H      ;Baud=28,800 bps
;RELOAD_COUNT EQU 0C4H      ;Baud=19,200 bps
;RELOAD_COUNT EQU 088H      ;Baud=9,600 bps
;以上是 Fosc = 18.432MHz，1T 模式，SMOD=1 时，计算出的自动重装数和波特率
;*************************************************************************
```

```
;*******************************************************************************
;以下是 Fosc = 18.432MHz，1T 模式，SMOD=0 时，计算出的自动重装数和波特率

;RELOAD_COUNT EQU 0FFH      ;Baud=576,000 bps
;RELOAD_COUNT EQU 0FEH      ;Baud=288,000 bps
;RELOAD_COUNT EQU 0FDH      ;Baud=144,000 bps
;RELOAD_COUNT EQU 0FCH      ;Baud=115,200 bps
;RELOAD_COUNT EQU 0F6H      ;Baud=57,600 bps
;RELOAD_COUNT EQU 0ECH      ;Baud=38,400 bps
;RELOAD_COUNT EQU 0E2H      ;Baud=28,800 bps
;RELOAD_COUNT EQU 0D8H      ;Baud=19,200 bps
;RELOAD_COUNT EQU 0C4H      ;Baud=9,600 bps
;RELOAD_COUNT EQU 088H      ;Baud=4,800 bps

;以上是 Fosc = 18.432MHz，1T 模式，SMOD=0 时，计算出的自动重装数和波特率
;*******************************************************************************

;*******************************************************************************
;以下是 Fosc = 18.432MHz，12T 模式，SMOD=0 时，计算出的自动重装数和波特率

RELOAD_COUNT EQU 0FBH      ;Baud=9,600 bps
;RELOAD_COUNT EQU 0F6H      ;Baud=4,800 bps
;RELOAD_COUNT EQU 0ECH      ;Baud=2,400 bps
;RELOAD_COUNT EQU 0D8H      ;Baud=1,200 bps

;以上是 Fosc = 18.432MHz，12T 模式，SMOD=0 时，计算出的自动重装数和波特率
;*******************************************************************************

;*******************************************************************************
;以下是 Fosc = 18.432MHz，12T 模式，SMOD=1 时，计算出的自动重装数和波特率

;RELOAD_COUNT EQU 0FBH      ;Baud=19,200 bps
;RELOAD_COUNT EQU 0F6H      ;Baud=9,600 bps
;RELOAD_COUNT EQU 0ECH      ;Baud=4,800 bps
;RELOAD_COUNT EQU 0D8H      ;Baud=2,400 bps
;RELOAD_COUNT EQU 0B0H      ;Baud=1,200 bps

;以上是 Fosc = 18.432MHz，12T 模式，SMOD=1 时，计算出的自动重装数和波特率
;*******************************************************************************
```

```
;*******************************************************************************
;以下是 Fosc = 11.0592MHz，12T 模式，SMOD=0 时，计算出的自动重装数和波特率

;RELOAD_COUNT EQU 0FFH      ;Baud=28,800 bps
;RELOAD_COUNT EQU 0FEH      ;Baud=14,400 bps
;RELOAD_COUNT EQU 0FDH      ;Baud=9,600 bps
;RELOAD_COUNT EQU 0FAH      ;Baud=4,800 bps
;RELOAD_COUNT EQU 0F4H      ;Baud=2,400 bps
;RELOAD_COUNT EQU 0E8H      ;Baud=1,200 bps

;以上是 Fosc = 11.0592MHz，12T 模式，SMOD=0 时，计算出的自动重装数和波特率
;*******************************************************************************

;*******************************************************************************
;以下是 Fosc = 11.0592MHz，12T 模式，SMOD=1 时，计算出的自动重装数和波特率

;RELOAD_COUNT EQU 0FFH      ;Baud=57,600 bps
;RELOAD_COUNT EQU 0FEH      ;Baud=28,800 bps
;RELOAD_COUNT EQU 0FDH      ;Baud=14,400 bps
;RELOAD_COUNT EQU 0FAH      ;Baud=9,600 bps
;RELOAD_COUNT EQU 0F4H      ;Baud=4,800 bps
;RELOAD_COUNT EQU 0E8H      ;Baud=2,400 bps
;RELOAD_COUNT EQU 0D0H      ;Baud=1,200 bps

;以上是 Fosc = 11.0592MHz，12T 模式，SMOD=1 时，计算出的自动重装数和波特率
;*******************************************************************************

;定义指示灯
LED_MCU_START  EQU  P1.7           ;MCU 工作指示灯
;-------------------------------------------------------------------------------
     ORG   0000H
    AJMP  MAIN
;-------------------------------------------------------------------------------
    ORG   0023H
    AJMP  UART_Interrupt           ;RS232 串口中断服务程序
    NOP
    NOP
;-------------------------------------------------------------------------------
MAIN:
    MOV   SP, #7FH                 ;设置堆栈指针
    CLR   LED_MCU_START            ;点亮 MCU 工作指示灯
    ACALL Initial_UART             ;初始化串口
    MOV   R0, #30H                 ;30H = 可打印字符 '0' 的 ASCII 码
    MOV   R2, #10                  ;发送 10 个字符 '0123456789'
```

```
LOOP:
    MOV   A, R0
    ACALL Send_One_Byte             ;发送一个字节,可将 PC 串口调试助手设置成字符显示
    ;如果是字符显示，显示为 0123456789,
    ;如设置成 16 进制显示，显示 30 31 32 33 34 35 36 37 38 39
    INC   R0
    DJNZ  R2, LOOP
MAIN_WAIT:
    SJMP  MAIN_WAIT                 ;跳转到本行，无限循环
;--------------------------------------------------------------------------------
UART_Interrupt:                     ;串口中断服务程序
    JB    RI, Is_UART_Receive
    CLR   TI                        ;清零串口发送中断标志
    RETI                             ;发送时使用的是查询方式，不使用中断
Is_UART_Receive:
    CLR   RI
    PUSH  ACC
    MOV   A, SBUF                   ;取接收到的字节
    ACALL Send_One_Byte             ;回发收到的字节
    POP   ACC
    RETI
;--------------------------------------------------------------------------------
Initial_UART:                       ;初始化串口
; SCON  Bit:   7       6     5     4     3     2     1    0
;           SM0/FE   SM1   SM2   REN   TB8   RB8    TI   RI
    MOV   SCON, #50H                ; 0101,0000 8 位可变波特率，无奇偶校验

    MOV   TMOD, #21H                ;设置定时器 1 为 8 位自动重装计数器
    MOV   TH1, #RELOAD_COUNT        ;设置定时器 1 自动重装数
    MOV   TL1, #RELOAD_COUNT

;---------------------------------------
;    ORL   PCON, #80H               ;若本行有效，波特率可以加倍
;---------------------------------------
;以下两行指令只能有一行有效
;    ORL  AUXR, #01000000B           ;定时器 1 工作在 1T 模式，波特率可以快 12 倍
     ANL  AUXR, #10111111B          ;定时器 1 工作在 12T 模式，与普通的 8051 相同
;以上两行指令只能有一行有效
;---------------------------------------
    SETB  TR1                       ;启动定时器 1
    SETB  ES
    SETB  EA
    RET
```

```
;-------------------------------------------------------------------------
;入口参数: A = 要发送的字节
Send_One_Byte:                      ;发送一个字节
    CLR   ES
    CLR   TI                        ;清零串口发送中断标志
    MOV   SBUF, A
Wait_Send_Finish:
    JNB   TI, Wait_Send_Finish      ;等待发送完毕
    CLR   TI                        ;清零串口发送中断标志
    SETB  ES
    RET
;-------------------------------------------------------------------------
    END
;-------------------------------------------------------------------------
;计算自动重装数 RELOAD (SMOD = 0, SMOD 是 PCON 特殊功能寄存器的最高位):
;  1. 计算 RELOAD (以下是 SMOD = 0 时的计算公式)
;
;     a) 12T 模式的计算公式： RELOAD = 256 - INT(Fosc/Baud0/32/12 + 0.5)
;     b)  1T 模式的计算公式： RELOAD = 256 - INT(Fosc/Baud0/32 + 0.5)
;
;     式中: INT() 表示取整运算即舍去小数, 在式中加 0.5 可以达到四舍五入的目的
;           Fosc = 晶振频率
;           Baud0 = 标准波特率
;
;  2. 计算用 RELOAD 产生的波特率:
;       a)    Baud = Fosc/(256 - RELOAD)/32/12       12T 模式
;       b)    Baud = Fosc/(256 - RELOAD)/32           1T 模式
;
;  3. 计算误差
;          error = (Baud - Baud0)/Baud0 * 100%
;  4. 如果误差绝对值 > 3% 要更换波特率或者更换晶体频率, 重复步骤 1-4
;
;
;例: Fosc = 22.1184MHz, Baud0 = 57600 (12T 模式)
; 1. RELOAD = 256 - INT( 22118400/57600/32/12 + 0.5)
;           = 256 - INT( 1.5 )
;           = 256 - 1
;           = 255
;           = 0FFH
; 2. Baud = 22118400/(256-255)/32/12
;         = 57600
; 3. 误差等于零
```

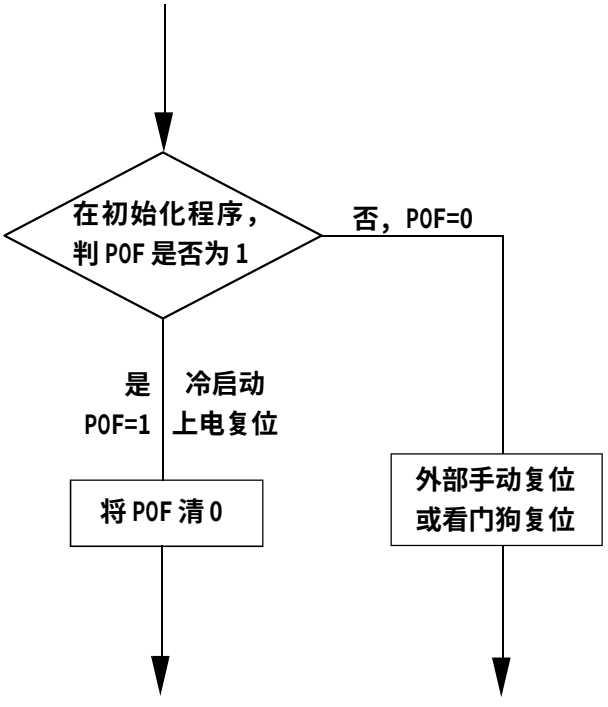
```
;例: Fosc = 18.432MHz, Baud0 = 57600 (12T 模式)
; 1. RELOAD = 256 - INT( 18432000/57600/32/12 + 0.5)
;           = 256 - INT( 0.833 + 0.5 )
;           = 256 - INT( 1.333 )
;           = 256 - 1
;           = 255
;           = 0FFH
; 2. Baud = 18432000/(256-255)/32/12
;         = 48000
; 3. error = (48000 - 57600)/57600 * 100%
;          = -16.66%
; 4. 误差很大，要更换波特率或者更换晶体频率，重新计算请见下一例

;例: Fosc = 18.432MHz, Baud0 = 9600 (12T 模式)
; 1. RELOAD = 256 - INT( 18432000/9600/32/12 + 0.5)
;           = 256 - INT( 5.5 )
;           = 256 - 5
;           = 251
;           = 0FBH
; 2. Baud = 18432000/(256-251)/32/12
;         = 9600
; 3. 一目了然，误差等于零

;例: Fosc = 2.000MHz, Baud = 4800 (1T 模式)
; 1. RELOAD = 256 - INT( 2000000/4800/32 + 0.5)
;           = 256 - INT( 13.02 + 0.5 )
;           = 256 - INT( 13.52 )
;           = 256 - 13
;           = 243
;           = 0F3H
; 2. Baud = 2000000/(256-243)/32
;         = 4808
; 3. error = 0.16%
;-------------------------------------------------------------------------------
```

第六章　　STC11/10xx 系列单片机的省电模式

6.1　PCON 寄存器的高级应用，上电复位标志
如何进入掉电模式和空闲模式

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset value
PCON	87h	Power Control	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000

POF：上电复位标志位，单片机停电后，上电复位标志位为 1，可由软件清 0。
实际应用：要判断是上电复位（冷启动），还是外部复位脚输入复位信号产生的复位，还是内部看门狗复位，可通过如下方法来判断：



PD：将其置 1 时，进入 Power Down 模式，可由外部中断低电平触发或下降沿触发中断模式唤醒,也可启动掉电唤醒专用定时器唤醒。
进入掉电模式时，外部时钟停振，CPU、定时器、串行口全部停止工作，只有外部中断继续工作。
可将 CPU 从掉电模式唤醒的外部管脚有：INT0/P3.2，INT1/P3.3，INT/T0/P3.4，INT/T1/P3.5，INT/RxD/P3.0（或 INT/RxD/P1.6）
INT0/P3.2,INT1/P3.3: 支持下降沿和低电平唤醒,由外部中断口的中断模式决定
INT/T0/P3.4，INT/T1/P3.5,INT/RxD/P3.0（或 INT/RxD/P1.6):支持下降沿唤醒

IDL：将其置 1，进入 IDLE 模式（空闲），除 CPU 不工作外，其余仍继续工作，可由任何一个中断唤醒。
可将 CPU 从空闲模式(IDLE 模式)唤醒的外部中断脚有:
INT0/P3.2，INT1/P3.3，INT/T0/P3.4,INT/T1/P3.5,INT/RxD/P3.0（或 INT/RxD/P1.6）
内部定时器 Timer0,Timer1 也可以将单片机从空闲模式唤醒
串行口中断(UART)也可以将单片机从空闲模式唤醒

GF1,GF0:　两个通用工作标志位,用户可以任意使用。
SMOD：　波特率倍速位，置 1，串口通讯波特率快一倍

6.2 利用外部中断实现单片机从掉电模式唤醒(C语言)

```
/* ---  STC International Limited ------------------------------------------- */
/* ---  宏晶科技     姚永平     2006/8/2     V1.0 ---------------------------------- */
/* ---  STC11/10xx 系列单片机,掉电模式唤醒测试程序(从外部中断0唤醒)----------------*/
/* ---  本演示程序在STC-ISP Ver 3.0A.PCB的下载编程工具上测试通过 ----------------- */
/* ---  如果要在程序中使用该程序,请在程序中注明使用了宏晶科技的资料及程序 ----------*/
/* ---  如果要在文章中引用该程序,请在文章中注明使用了宏晶科技的资料及程序 ---------- */

#include<reg51.h>
#include<intrins.h>

sbit  Begin_Led = P1^2;           //系统开始工作指示灯
unsigned char Is_Power_Down = 0; //进入Power Down 之前，将其置为1，以供判断
sbit  Is_Power_Down_Led_INT0     = P1^7; //掉电唤醒指示灯，在外部中断0中
sbit  Not_Power_Down_Led_INT0    = P1^6; //不是掉电唤醒指示灯，在外部中断0中
sbit  Is_Power_Down_Led_INT1     = P1^5; //掉电唤醒指示灯，在外部中断1中
sbit  Not_Power_Down_Led_INT1    = P1^4; //不是掉电唤醒指示灯，在外部中断1中
sbit  Power_Down_Wakeup_Pin_INT0 = P3^2; //掉电唤醒管脚，外部中断0
sbit  Power_Down_Wakeup_Pin_INT1 = P3^3; //掉电唤醒管脚，外部中断1
sbit  Normal_Work_Flashing_Led   = P1^3; //系统处于正常工作状态指示灯


void Normal_Work_Flashing(void);
void INT_System_init(void);
void INT0_Routine(void);
void INT1_Routine(void);

void main(void)
{
    unsigned  char  j  =  0;
    unsigned  char  wakeup_counter  =  0;  //中断唤醒次数变量初始为0
    Begin_Led  =  0;       //系统开始工作指示灯
    INT_System_init();    //中断系统初始化
    while(1)
    {
        P2  = ~wakeup_counter;    //中断唤醒次数显示,先将wakeup_counter取反
        wakeup_counter++;         //中断唤醒次数显示
        for(j=0;j<2;j++)
        {
              Normal_Work_Flashing();    //系统正常工作指示灯
        }
         Is_Power_Down  =  1; //进入Power Down 之前，将其置为1，以供判断
         PCON  =  0x02; //执行完此句，单片机进入Power Down模式，外部时钟停止振荡
```

```
        _nop_();
        //STC12系列掉电模式，外部中断唤醒后，首先执行上句，然后才会进入中断服务程序
        _nop_();
        _nop_();  //建议多加几个空操作指令NOP
        _nop_();  //建议多加几个空操作指令NOP
    }
}

void INT_System_init(void)
{
    IT0   =   0;   /* 外部中断0，低电平触发中断 */
//  IT0   =   1;   /* 外部中断0，下降沿触发中断 */
    EX0   =   1;   /* 允许外部中断0中断 */
    IT1   =   0;   /* 外部中断1，低电平触发中断 */
//  IT1   =   1;   /* 外部中断1，下降沿触发中断 */
    EX1   =   1;   /* 允许外部中断1中断 */
    EA    =   1;   /* 开总中断控制位 */
}

void INT0_Routine(void) interrupt 0
{
    if(Is_Power_Down)
    {   //Is_Power_Down ==1,掉电唤醒,在外部中断0中
        Is_Power_Down  =   0;
        Is_Power_Down_Led_INT0  =  0;  //点亮外部中断0掉电唤醒指示灯
        while(Power_Down_Wakeup_Pin_INT0==0)
        {
            /* 等待变高 */
        }
        Is_Power_Down_Led_INT0  =  1;  //关闭外部中断0掉电唤醒指示灯
    }
    else
    {
        Not_Power_Down_Led_INT0 =  0;  //点亮外部中断0正常工作中断指示灯
        while(Power_Down_Wakeup_Pin_INT0==0)
        {
            /* 等待变高 */
        }
        Not_Power_Down_Led_INT0 =  1;  //关闭外部中断0正常工作中断指示灯
    }
}

void INT1_Routine(void) interrupt 2
{
    if(Is_Power_Down)
    {   //Is_Power_Down ==1,掉电唤醒,在外部中断1中
        Is_Power_Down   =   0;
        Is_Power_Down_Led_INT1 =  0;  //顶亮外部中断1掉电唤醒指示灯
```

```
        while(Power_Down_Wakeup_Pin_INT1==0)
        {
            /* 等待变高 */
        }
        Is_Power_Down_Led_INT1  =  1;  // 关闭外部中断 1 掉电唤醒指示灯
    }
    else
    {
        Not_Power_Down_Led_INT1  =  0;  // 顶亮外部中断 1 正常工作中断指示灯
        while(Power_Down_Wakeup_Pin_INT1==0)
        {
            /* 等待变高 */
        }
        Not_Power_Down_Led_INT1  =  1;  // 关闭外部中断 1 正常工作中断指示灯
    }
}
void delay(void)
{
    unsigned int    j   =   0x00;
    unsigned int    k   =   0x00;
    for(k=0;k<2;++k)
    {
        for(j=0;j<=30000;++j)
        {
             _nop_();
             _nop_();
             _nop_();
             _nop_();
             _nop_();
             _nop_();
             _nop_();
             _nop_();
        }
    }
}
void Normal_Work_Flashing(void)
{
    Normal_Work_Flashing_Led    =   0;
    delay();
    Normal_Work_Flashing_Led    =   1;
    delay();
}
```

6.3 通过外部中断从掉电模式唤醒

```
;*************************************************************************
;Wake Up Idle and Wake Up Power Down
;*************************************************************************
    ORG   0000H
    AJMP  MAIN

    ORG   0003H
int0_interrupt:
    CLR   P1.7          ;点亮 P1.7 LED 表示已响应 int0 中断
    ACALL delay         ;延时是为了便于观察，实际应用不需延时
    CLR   EA            ;关闭中断，简化实验．实际应用不需关闭中断
    RETI

    ORG   0013H
int1_interrupt:
    CLR   P1.6          ;点亮 P1.6 LED 表示已响应 int1 中断
    ACALL delay         ;延时是为了便于观察，实际应用不需延时
    CLR   EA            ;关闭中断，简化实验．实际应用不需关闭中断
    RETI

    ORG   0100H
delay:
    CLR   A
    MOV   R0, A
    MOV   R1, A
    MOV   R2, #02
delay_loop:
    DJNZ  R0, delay_loop
    DJNZ  R1, delay_loop
    DJNZ  R2, delay_loop
    RET

main:
    MOV   R3, #0        ;P1 LED 递增方式变化，表示程序开始运行
main_loop:
    MOV   A, R3
    CPL   A
    MOV   P1, A
```

```
    INC   R3
    MOV   A, R3
    SUBB  A, #18H
    JC    main_loop

    MOV   P1, #0FFH     ;熄灭全部灯表示进入 Power Down 状态

    CLR   IT0           ;设置低电平激活外部中断
;   SETB  IT0

    SETB  EX0           ;允许外部中断 0

    CLR   IT1           ;设置低电平激活外部中断
;   SETB  IT1
    SETB  EX1           ;允许外部中断 1

    SETB  EA            ;开中断, 若不开中断就不能唤醒 Power Down

;下条语句将使 MCU 进入 idle 状态或 Power Down 状态
;低电平激活外部中断可以将 MCU 从 Power Down 状态中唤醒
;其方法为:将外部中断脚拉低

    MOV   PCON, #00000010B     ;令 PD=1, 进入 Power Down 状态, PD = PCON.1
;    NOP       ;实际使用掉电模式时,应在 MOV   PCON, #00000010B 语句后面多加几个 NOP
;    NOP       ;实际使用掉电模式时,应在 MOV   PCON, #00000010B 语句后面多加几个 NOP
;    NOP       ;实际使用掉电模式时,应在 MOV   PCON, #00000010B 语句后面多加几个 NOP

    ;MOV   PCON, #00000001B    ;删除本语句前的 ";", 同时将前 1 条语句前加上注释符号 ";",
                              ;令 IDL=1, 可进入 idle 状态, IDL = PCON.0

    MOV   P1, #0DFH     ;1101,1111  请注意:
                        ; 1.外部中断使 MCU 退出 Power Down 状态,执行本条指令后
                        ;响应中断, 表现为 P1.5 与 P1.7 的 LED 同时亮(INT0 唤醒)
                        ; 2.外部中断使 MCU 退出 idle 状态,先响应中断然后再执行本
                        ;条指令, 表现为 P1.7 的 LED 先亮(INT0 唤醒)P1.5 的 LED 后亮
                        ; 3.实际使用掉电模式时,本语句应用 NOP 代替
    NOP       ;实际使用掉电模式时,应在 MOV   PCON, #00000010B 语句后面多加几个 NOP
    NOP       ;实际使用掉电模式时,应在 MOV   PCON, #00000010B 语句后面多加几个 NOP
    NOP       ;实际使用掉电模式时,应在 MOV   PCON, #00000010B 语句后面多加几个 NOP
WAIT1:
    SJMP  WAIT1         ;跳转到本语句, 停机
    END
```

6.4 进入掉电模式后,由内部掉电唤醒专用定时器唤醒的应用说明

STC11xx 系列单片机在进入掉电模式后，除了可以通过外部中断源进行唤醒外，还可以在无外部中断源的情况下通过使能内部掉电唤醒定时器定期唤醒CPU，使其恢复到正常工作状态(STC10xx系列无此功能)。

WKTCL 地址：AAH()

7	6	5	4	3	2	1	0	Reset Value
								0000,0000

WKTCH 地址：ABH

7	6	5	4	3	2	1	0	Reset Value
WKTEN	-	-	-					0xxx,0000

{WKTCH[3:0],WKTCL[7:0]}构成最长 12 位计数值(4096 个)

通过软件将 WKTCH 寄存器中的 WKTEN(Power Down Wakeup Timer Enable)位置 ‘1’,使能内部掉电唤醒专用定时器,当MCU一旦进入Power Down Mode,内部掉电唤醒专用定时器就开始计数,直到计数到与{WKTCH[3:0],WKTCL[7:0]}寄存器所设定的计数值相等后就启动系统振荡器,MCU等待32768/16384/8192/4096个时钟(由用户在ISP烧录程序时自行设置)后,MCU认为此时系统时钟从开始起振的不稳定状态已经过渡到稳定状态,才将时钟供给 CPU 工作，CPU 获得时钟后，程序从上次掉电的地方继续往下执行。

内部定时器计数一次的时间约为 560us,当然误差较大。

内部掉电唤醒专用定时器最短计数时间约为 560uS

内部掉电唤醒专用定时器最长计数时间约为 560us x 4096 = 2.3S

例如： {设定 WKTCH[3:0],WKTCL[7:0]}寄存器的值等于 10，则从系统掉电到启动系统振荡器，所需要等待的时间为 560uS x 10 = 5600uS

设定{WKTCH[3:0],WKTCL[7:0]}寄存器的值等于 4096(最大值 = 4096 = 2^{12})，则从系统掉电到启动系统振荡器，所需要等待的时间为 560uS x 4096 = 2.3 S

{WKTCH[3:0],WKTCL[7:0]} = 1, 560uS x 1 = 560uS
{WKTCH[3:0],WKTCL[7:0]} = 10, 560uS x 10 = 5.6mS
{WKTCH[3:0],WKTCL[7:0]} = 100, 560uS x 100 = 56mS
{WKTCH[3:0],WKTCL[7:0]} = 1000, 560uS x 1000 = 560mS
{WKTCH[3:0],WKTCL[7:0]} = 4096, 560uS x 4096 = 2.3S

掉电模式功耗：单片机在掉电模式下的典型功耗为 2uA。

第七章　STC11/10xx系列单片机电气特性

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

Parameter	Symbol	MIN	MAX	UNIT
Storage temperature	T_{ST}	-55	+125	℃
Operating Temperature(I)	T_A	-40	+85	℃
Operating Temperature(C)	T_A	0	+70	℃
DC Power Supply(5V MCU)	V_{DD} - V_{SS}	-0.3	+5.5	V
DC Power Supply(3V MCU)	V_{DD} - V_{SS}	-0.3	+3.6	V
Voltage on any Pin		-0.3	VCC + 0.3	V

DC Specification(5V MCU)

Symbol	Parameter	Specification				Test Condition
		Min.	Typ.	Max.	Unit	
V_{DD}	Operating Voltage	4.1	5.0	5.5	V	
I_{PWDN}	Power Down Current		<0.1		uA	5V
I_{IDLE}	Idle Current		3.0		mA	5V
I_{CC}	Operating Current		4 mA	20	mA	5V
V_{IL1}	Input low voltage (P0,P1,P2,P3)			0.8	V	5V
V_{IH1}	Input High voltage (P0,P1,P2,P3)	2.0			V	5V
V_{IH2}	Input High voltage (RESET)	2.2			V	5V
I_{OL1}	Sinking Current for Output Low (P0,P1,P2,P3)		20		mA	5V Vpin=0.45V
I_{OH1}	(Quasi-output) Sourcing Current for Output high (P0,P1,P2,P3)	150	230		uA	5V
I_{OH2}	(Push-Pull,Strong-output) Sourcing Current for Output High (P0,P1,P2,P3)		20		mA	5V Vpin=2.4V
I_{IL}	Logic 0 input current (P0,P1,P2,P3)			50	uA	V_{PIN}=0V
I_{TL}	Logic 1 to 0 transition current (P0,P1,P2,P3)	100	270	600	uA	V_{PIN}=2V

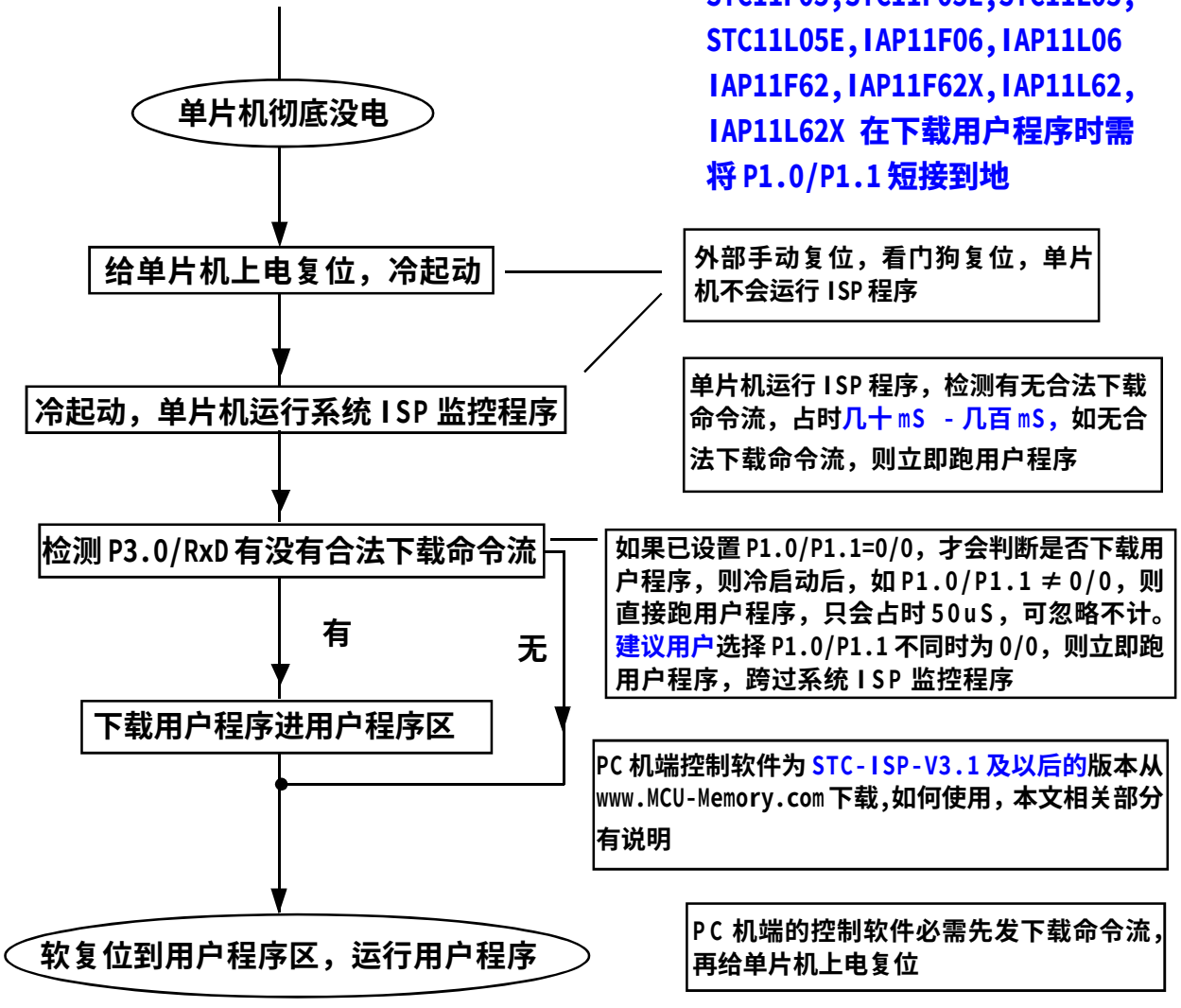
DC Specification(3.3V MCU)

Symbol	Parameter	Specification				Test Condition
		Min.	Typ.	Max.	Unit	
V_{DD}	Operating Voltage	2.2	3.3	3.6	V	
I_{PWDN}	Power Down Current		<0.1		uA	3.3V
I_{IDLE}	Idle Current		2.0		mA	3.3V
I_{CC}	Operating Current		4 mA	10	mA	3.3V
V_{IL1}	Input low voltage (P0,P1,P2,P3)			0.8	V	3.3V
V_{IH1}	Input High voltage (P0,P1,P2,P3)	2.0			V	3.3V
V_{IH2}	Input High voltage (RESET)	2.2			V	3.3V
I_{OL1}	Sinking Current for Output Low (P0,P1,P2,P3)		20		mA	3.3V Vpin=0.45V
I_{OH1}	(QUasi-output) Sourcing Current for Output High (P0,P1,P2,P3)	40	70		uA	3.3V
I_{OH2}	(Push-Pull,Strong-output) Sourcing Current for Output High (P0,P1,P2,P3)		20		mA	3.3V
I_{IL}	Logic 0 input current (P0,P1,P2,P3)		8	50	uA	V_{PIN}=0V
I_{TL}	Logic 1 to 0 transition current (P0,P1,P2,P3)		110	600	uA	V_{PIN}=2V

第八章　STC11/10xx 系列单片机开发 / 编程工具说明

8.1 在系统可编程（ISP）原理，官方演示工具使用说明

8.1.1 在系统可编程（ISP）原理使用说明



STC11F05,STC11F05E,STC11L05,
STC11L05E,IAP11F06,IAP11L06
IAP11F62,IAP11F62X,IAP11L62,
IAP11L62X 在下载用户程序时需
将 P1.0/P1.1 短接到地

外部手动复位，看门狗复位，单片机不会运行 ISP 程序

单片机运行 ISP 程序，检测有无合法下载命令流，占时几十 mS - 几百 mS，如无合法下载命令流，则立即跑用户程序

如果已设置 P1.0/P1.1=0/0，才会判断是否下载用户程序，则冷启动后，如 P1.0/P1.1 ≠ 0/0，则直接跑用户程序，只会占时 50uS，可忽略不计。建议用户选择 P1.0/P1.1 不同时为 0/0，则立即跑用户程序，跨过系统 ISP 监控程序

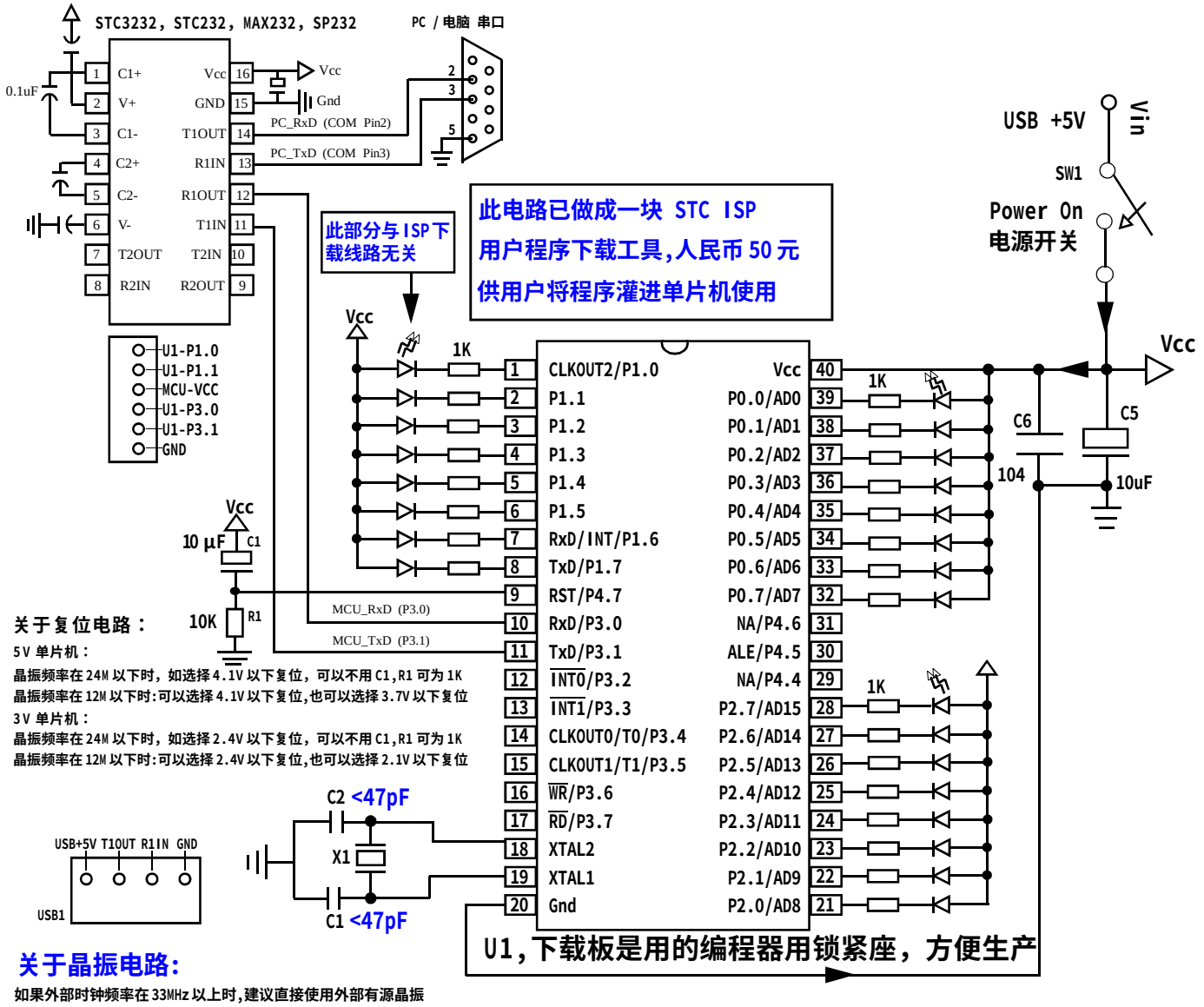
PC 机端控制软件为 STC-ISP-V3.1 及以后的版本从 www.MCU-Memory.com 下载,如何使用，本文相关部分有说明

PC 机端的控制软件必需先发下载命令流，再给单片机上电复位

8.1.2 STC11/10xx 系列在系统可编程（ISP）典型应用线路图

串行口做主机通信时，可控制串口通信在[RxD/P3.0,TxD/P3.1]和[RxD/P1.6,TxD/P1.7.]之间任意切换，实现 2 组串口。建议用户将自己的串行口设置在[RxD/P1.6,TxD/P1.7.]而将[RxD/P3.0,TxD/P3.1]口作为 ISP 下载的专用通信口,当然也可以当用户的普通 I/O 口用

STC 单片机在线编程线路， *STC* RS-232 转换器



关于复位电路：

5V 单片机：

晶振频率在 24M 以下时，如选择 4.1V 以下复位，可以不用 C1,R1 可为 1K

晶振频率在 12M 以下时:可以选择 4.1V 以下复位,也可以选择 3.7V 以下复位

3V 单片机：

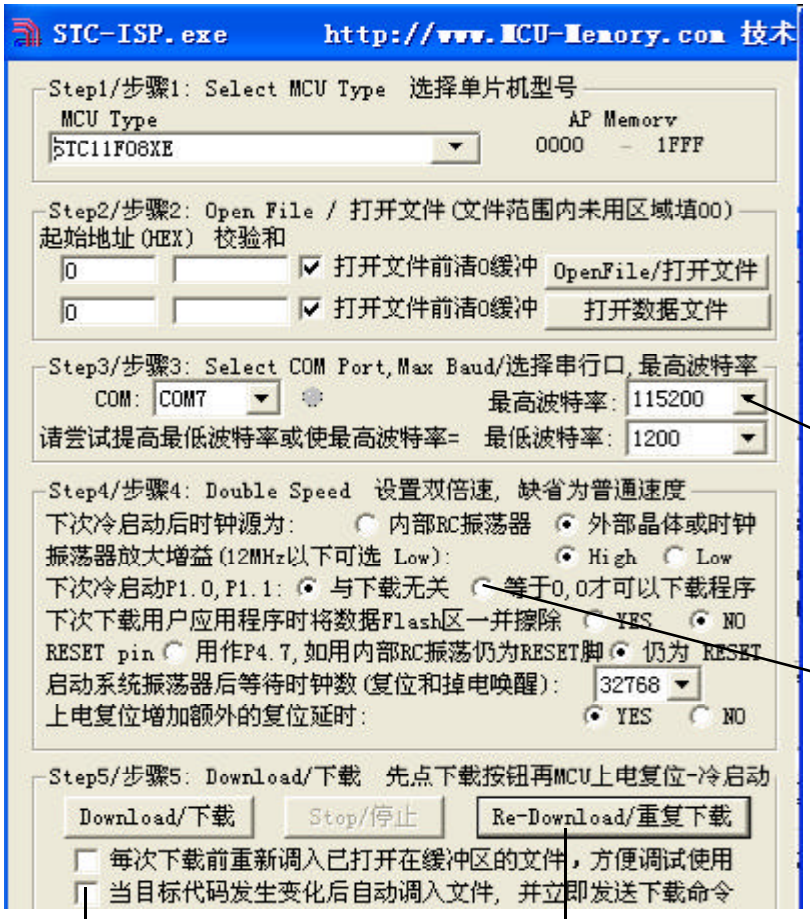
晶振频率在 24M 以下时，如选择 2.4V 以下复位，可以不用 C1,R1 可为 1K

晶振频率在 12M 以下时:可以选择 2.4V 以下复位,也可以选择 2.1V 以下复位

关于晶振电路:

如果外部时钟频率在 33MHz 以上时,建议直接使用外部有源晶振

8.1.3 电脑端的 ISP 控制软件界面使用说明



用户根据实际使用效果选择限制最高通信波特率，如 57600，38400，19200

如 P30/P31 外接 RS-485/RS-232 等通信电路，建议选择 P10/P11 等于 0/0 才可以下载程序，如不同时为 0/0，则跨过系统 ISP 引导程序，直接运行用户程序。

新的设置冷启动后（彻底停电后再上电），才生效

开发调试时，可以考虑选择此项

大批量生产时使用

Step1/ 步骤 1：选择你所使用的单片机型号，如 STC11Fxx 等

Step2/ 步骤 2：打开文件，要烧录用户程序，必须调入用户的程序代码（*.bin， *.hex）

Step3/ 步骤 3：选择串行口，你所使用的电脑串口，如串行口 1--COM1， 串行口 2--COM2,...

有些新式笔记本电脑没有 RS-232 串行口,可买一条 USB-RS232 转接器，人民币 50 元左右。

有些 USB-RS232 转接器，不能兼容，可让宏晶帮你购买经过测试的转换器。

Step4/ 步骤 4：选择下次冷启动后，时钟源为“内部 R/C 振荡器”还是“外部晶体或时钟”。

Step5/ 步骤 5：选择“Download/ 下载”按钮下载用户的程序进单片机内部，可重复执行

Step5/ 步骤 5， 也可选择“Re-Download/ 重复下载”按钮

下载时注意看提示，主要看是否要给单片机上电或复位， 下载速度比一般通用编程器快。

一定要先选择“Download/ 下载”按钮，然后再给单片机上电复位(先彻底断电)，而不要先上电，先上电，检测不到合法的下载命令流，单片机就直接跑用户程序了。

关于硬件连接：

(1). MCU/ 单片机 RXD(P3.0) --- RS-232 转换器 --- PC/ 电脑 TXD(COM Port Pin3)

(2). MCU/ 单片机 TXD(P3.1) --- RS-232 转换器 --- PC/ 电脑 RXD(COM Port Pin2)

(3). MCU/ 单片机 GND ---------- PC/ 电脑 GND(COM Port Pin5)

(4). 如果您的系统 P3.0/P3.1 连接到 RS-485 电路，推荐

在选项里选择“下次冷启动需要 P1.0/P1.1 = 0,0 才可以下载用户程序”

这样冷启动后如 P1.0，P1.1 不同时为 0,单片机直接运行用户程序，免得由于 RS-485 总线上的乱码造成单片机反复判断乱码是否为合法，浪费几百 mS 的时间，其实如果你的系统本身 P3.0,P3.1 就是做串口使用，也建议选择 P1.0/P1.1 = 0/0 才可下载用户程序，以便下次冷启动直接运行用户程序。

(5). RS-232 转换器可选用 MAX232/SP232(4.5-5.5V),MAX3232/SP3232(3V-5.5V).

8.1.4 宏晶科技的 ISP 下载编程工具硬件使用说明

如用户系统没有 RS-232 接口，
可使用STC-ISP Ver 3.0A.PCB演示板作为编程工具

STC-ISP Ver 3.0APCB 板可以焊接 3 种电路，分别支持 STC12 系列 16Pin / 20Pin / 28Pin / 32Pin。我们在下载板的反面贴了一张标签纸，说明它是支持 16Pin /20Pin / 28Pin / 32Pin 中的哪一种，用户要特别注意。在正面焊的编程烧录用锁紧座都是 40Pin 的，锁紧座第 20-Pin 接的是地线，请将单片机的地线对着锁紧座的地线插。

在 STC-ISP Ver 3.0A PCB 板完成下载编程用户程序的工作：

关于硬件连接：

(1). 根据单片机的工作电压选择单片机电源电压

A. 5V 单片机,短接 JP1 的 MCU-VCC，+5V 电源管脚

B. 3V 单片机,短接 JP1 的 MCU-VCC，3.3V 电源管脚

(2).连接线(宏晶提供)

A. 将一端有 9 芯连接座的插头插入 PC/ 电脑 RS-232 串行接口插座用于通信

B. 将同一端的 USB 插头插入 PC/ 电脑 USB 接口用于取电

C. 将只有一个 USB 插头的一端插入宏晶的 STC-ISP Ver 3.0A PCB 板 USB1 插座用于 RS-232 通信和供电,此时 USB +5V Power 灯亮(D43,USB 接口有电)

(3).其他插座不需连接

(4).SW1 开关处于非按下状态，此时 MCU-VCC Power 灯不亮(D41)，没有给单片机通电

(5).SW3 开关

处于非按下状态，P1.0，P1.1 = 1，1,不短接到地。

处于按下状态， P1.0，P1.1 = 0，0,短接到地。

如果单片机已被设成“下次冷启动 P1.0/P1.1 = 0,0 才判 P3.0/RxD 有无合法下载命令流”

就必须将 SW3 开关处于按下状态，让单片机的 P1.0/P1.1 短接到地

(6).将单片机插进 U1-Socket 锁紧座，锁紧单片机,注意单片机是 20-Pin / 28-Pin,而 U1-Socket 锁紧座是 40-Pin,我们的设计是靠下插，靠近晶体的那一端插。

(7).关于软件：选择“Download/ 下载”(必须在给单片机上电之前让 PC 先发一串合法下载命令)

(8).按下 SW1 开关，给单片机上电复位，此时 MCU-VCC Power 灯亮(D41)

此时 STC 单片机进入 ISP 模式(STC12 系列冷启动进入 ISP)

(9).下载成功后，再按 SW1 开关，此时 SW1 开关处于非按下状态，MCU-VCC Power 灯不亮(D41)，给单片机断电，取下单片机，换上新的单片机。

8.1.5 用户板没有 RS-232 转换器,如何用宏晶科技的 ISP 下载板做 RS-232 通信转换

利用STC-ISP Ver 3.0A PCB 板进行RS-232转换
单片机在用户自己的板上完成下载 / 烧录：

1. U1-Socket 锁紧座不得插入单片机
2. 将用户系统上的电源(MCU-VCC,GND)及单片机的 P3.0/RXD,P3.1/TXD 接入转换板 CN2 插座
   这样用户系统上的单片机就具备了与 PC/ 电脑进行通信的能力
3. 将用户系统的单片机的 P1.0,P1.1 接入转换板 CN2 插座(如果需要的话)
4. 如须 P1.0，P1.1 = 0，0,短接到地，可在用户系统上将其短接到地，或将 P1.0/P1.1 也从用户系统上引到 STC-ISP Ver3.0A PCB 板上，将 SW3 开关按下，则 P1.0/P1.1=0,0。
5. 关于软件：选择“Download/ 下载”
6. 给单片机系统上电复位(注意是从用户系统自供电，不要从电脑 USB 取电,电脑 USB 座不插)
7. 下载程序时，如用户板有外部看门狗电路，不得启动，单片机必须有正确的复位,但不能在 ISP 下载程序时被外部看门狗复位,如有，可将外部看门狗电路 WDI 端 / 或 WDO 端浮空
8. 如有 RS-485 晶片连到 P3.0/Rxd,P3.1/Txd,或其他线路，在下载时应将其断开。

8.2 编译器/汇编器，编程器，仿真器

STC 单片机应使用何种编译器/汇编器：

1.任何老的编译器/汇编器都可以支持，流行用 Keil C51

2.把 STC 单片机，当成 Intel 的 8052/87C52/87C54/87C58,Philips 的 P87C52/P87C54/P87C58 就可以了

3.如果要用到扩展的专用特殊功能寄存器，直接对该地址单元设置就行了，当然先声明特殊功能寄存器的地址较好

编程烧录器:

我们有: STC11Fxx/系列 ISP 经济型下载编程工具(人民币 50 元，可申请免费样品)

注意:有专门下载 28PIN/20PIN 的不同演示板，

28PIN 是 28PIN 的演示板,20PIN 是 20PIN 的演示板

仿真器:如您已有老的仿真器，可仿真普通 8052 的基本功能

STC11Fxx 系列单片机扩展功能如它仿不了

可以用 STC-ISP.EXE 直接下载用户程序看运行结果就可以了,如需观察变量,可自己写一小段测试程序通过串口输出到电脑端的 STC-ISP.EXE 的“串口调试助手”来显示,也很方便。

无须添加新的设备

无仿真器如何调试/开发用户程序

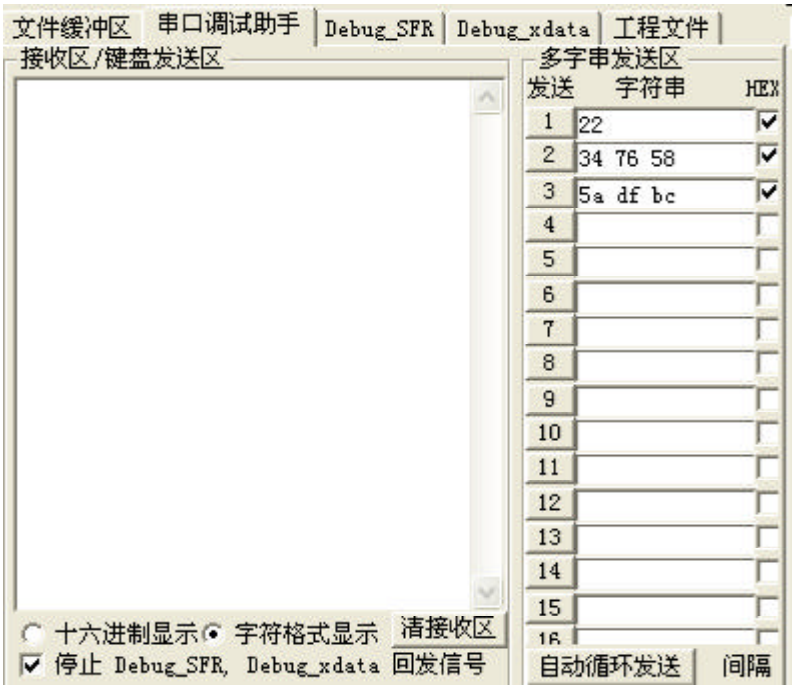
1.首先参照本手册当中的“用定时器 1 做波特率发生器”，调通串口程序，这样，要观察变量就可以自己写一小段测试程序将变量通过串口输出到电脑端的 STC-ISP.EXE 的“串口调试助手”来显示,也很方便 。

2.调通按键扫描程序(到处都有大量的参考程序)

3.调通用户系统的显示电路程序，此时变量/寄存器也可以通过用户系统的显示电路显示了

这样分步骤模块化调试用户程序，有些系统，熟练的 8051 用户，三天就可以调通了，难度不大的系统，一般一到二周就可以调通。

用户的串口输出显示程序可以在输出变量/寄存器的值之后，继续全速运行用户程序，也可以等待串口送来的“继续运行命令”，方可继续运行用户程序，这就相当于断点。这种断点每设置一个地方，就必须调用一次该显示寄存器 /变量的程序，有点麻烦，但却很实用。



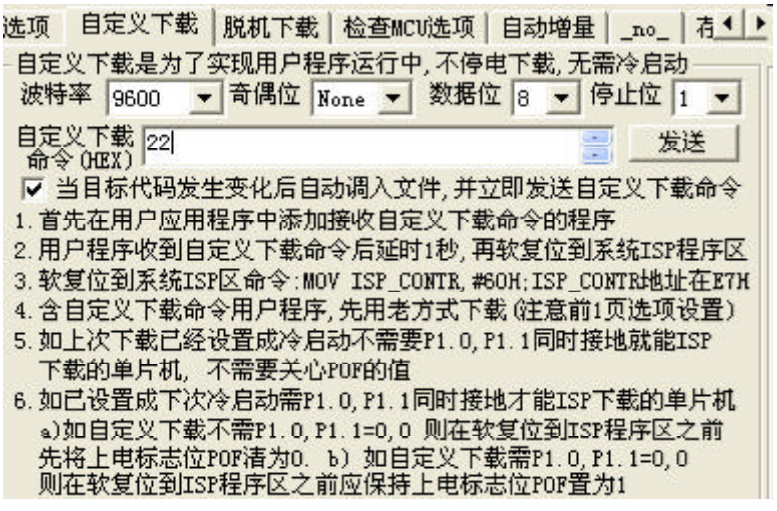
8.3 自定义下载演示程序(实现不停电下载)

```
/* ---  STC International Limited ------------------------------------------    */
/* ---  STC11/10xx  系列单片机,软件实现自定义下载程序 ---------------------------- */
/* ---  本演示程序在 STC-ISP Ver 3.0A.PCB 的下载编程工具上测试通过 ----------------  */
/* ---  如果要在程序中使用该程序,请在程序中注明使用了宏晶科技的资料及程序 ---------  */
/* ---  如果要在文章中引用该程序,请在文章中注明使用了宏晶科技的资料及程序 ----------  */
#include<reg51.h>
#include<intrins.h>
sfr  IAP_CONTR =  0xC7;
sbit MCU_Start_Led  =  P1^7;
//unsigned char self_command_array[4] =  {0x22,0x33,0x44,0x55};
#define Self_Define_ISP_Download_Command  0x22
#define RELOAD_COUNT  0xfb   //18.432MHz,12T,SMOD=0,9600bps
//#define RELOAD_COUNT  0xf6   //18.432MHz,12T,SMOD=0,4800bps
//#define RELOAD_COUNT  0xec   //18.432MHz,12T,SMOD=0,2400bps
//#define RELOAD_COUNT  0xd8   //18.432MHz,12T,SMOD=0,1200bps
void serial_port_initial();
void send_UART(unsigned char);
void UART_Interrupt_Receive(void);
void soft_reset_to_ISP_Monitor(void);
void delay(void);
void display_MCU_Start_Led(void);
void main(void)
{
    unsigned  char i = 0;
    serial_port_initial();      // 串口初始化
    display_MCU_Start_Led();    // 点亮发光二极管表示单片机开始工作
    send_UART(0x34);            // 串口发送数据表示单片机串口正常工作
    send_UART(0xa7);            // 串口发送数据表示单片机串口正常工作
    while(1);
}
void serial_port_initial()
{
    SCON    =   0x50;   //0101,0000 8 位可变波特率，无奇偶校验位
    TMOD    =   0x21;   //0011,0001 设置顶时器 1 为 8 位自动重装计数器
    TH1     =   RELOAD_COUNT;   // 设置定时器 1 自动重装数
    TL1     =   RELOAD_COUNT;
    TR1     =   1;    // 开定时器 1
    ES      =   1;    // 允许串口中断
    EA      =   1;    // 开总中断
}
```

```
void send_UART(unsigned char i)
{
    ES     =   0;  //关串口中断
    TI     =   0;  //清零串口发送完成中断请求标志
    SBUF   =   i;
    while(TI ==0); //等待发送完成
    TI     =   0;  //清零串口发送完成中断请求标志
    ES     =   1;  //允许串口中断
}

void UART_Interrupt_Receive(void) interrupt 4
{
    unsigned char   k   =   0;
    if(RI==1)
    {
        RI  =   0;
        k   =   SBUF;
        if(k==Self_Define_ISP_Download_Command)   //是自定义下载命令
        {
            delay();    //延时1秒就足够了
            delay();    //延时1秒就足够了
            soft_reset_to_ISP_Monitor();    //软复位到系统ISP监控区
        }
        send_UART(k);
    }
    else
    {
        TI  =  0;
    }
}

void soft_reset_to_ISP_Monitor(void)
{
    IAP_CONTR   =    0x60;   //0110,0000   软复位到系统ISP监控区
}
```

```
void delay(void)
{
    unsigned int j  =   0;
    unsigned int g  =   0;
    for(j=0;j<5;j++)
    {
        for(g=0;g<60000;g++)
        {
            _nop_();
            _nop_();
            _nop_();
            _nop_();
            _nop_();
        }
    }
}

void display_MCU_Start_Led(void)
{
    unsigned  char i = 0;
    for(i=0;i<3;i++)
    {
        MCU_Start_Led   =   0;  // 顶亮 MCU 开始工作指示灯
        delay();
        MCU_Start_Led   =   1;  // 熄灭 MCU 开始工作指示灯
        delay();
        MCU_Start_Led   =   0;  // 顶亮 MCU 开始工作指示灯
    }
}
```

自定义下载在 STC 的电脑端 ISP 软件 STC-ISP.EXE 中，还应做相应设置，具体参考设置见下图：



详细的帮助上图也有具体的说明

附录 A:　　内部常规 256 字节 RAM 间接寻址测试程序

```
;/* ---  STC International Limited ------------------------------------------ */
;/* ---  STC11/10xx 系列单片机 内部常规 RAM 间接寻址测试程序 ------------------- */

;/* ---  本演示程序在 STC-ISP Ver 3.0A.PCB 的下载编程工具上测试通过 ------------- */
;/* ---  如果要在程序中使用该程序,请在程序中注明使用了宏晶科技的资料及程序   ---- */
;/* ---  如果要在文章中引用该程序,请在文章中注明使用了宏晶科技的资料及程序   ---- */

TEST_CONST EQU   5AH
;TEST_RAM  EQU   03H
     ORG   0000H
      LJMP  INITIAL

      ORG   0050H
INITIAL:
     MOV   R0,   #253

      MOV   R1,   #3H
TEST_ALL_RAM:
      MOV   R2,   #0FFH
TEST_ONE_RAM:
      MOV   A,    R2
      MOV   @R1,  A
      CLR   A
      MOV   A,    @R1

      CJNE  A,    2H,   ERROR_DISPLAY
      DJNZ  R2,   TEST_ONE_RAM
      INC   R1
      DJNZ  R0,   TEST_ALL_RAM

OK_DISPLAY:
      MOV   P1,   #11111110B
Wait1:
     SJMP   Wait1

ERROR_DISPLAY:
      MOV   A,    R1
      MOV   P1,   A
Wait2:
     SJMP   Wait2
     END
```

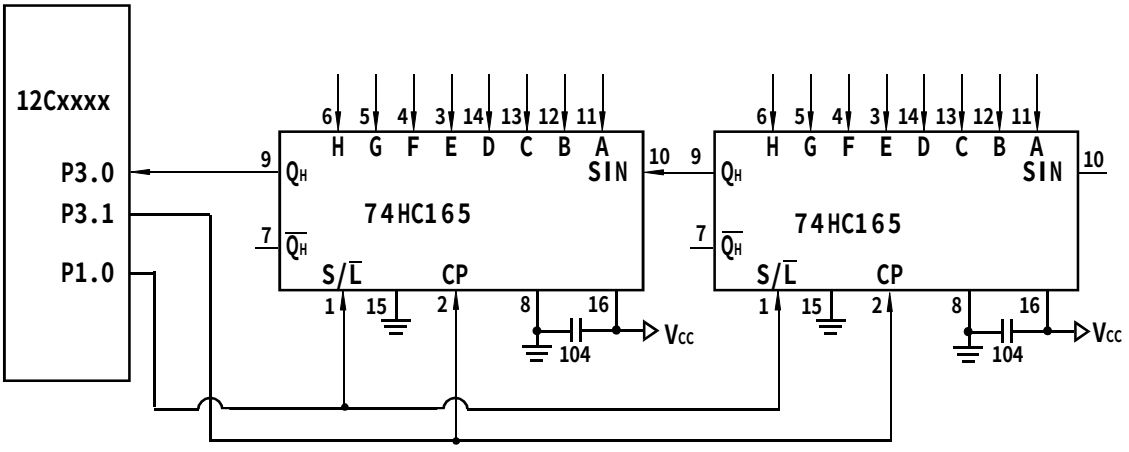
附录 B:　　用串行口扩展 I/O 接口

STC11/10xx 串行口的方式 0 可用于 I/O 扩展。如果在应用系统中，串行口未被占用，那么将它用来扩展并行 I/O 口是一种经济、实用的方法。

在操作方式 0 时，串行口作同步移位寄存器，其波特率是固定的，为 fosc/12（fosc 为振荡器频率）。数据由 RXD 端（P3.0）出入，同步移位时钟由 TXD 端（P3.1）输出。发送、接收的是 8 位数据，低位在先。

一、用 74HC165 扩展并行输入口

下图是利用两片 74HC165 扩展二个 8 位并行输入口的接口电路图。



74HC165 是 8 位并行置入移位寄存器。当移位 / 置入端(S/$\overline{L}$)由高到低跳变时，并行输入端的数据置入寄存器；当 S/$\overline{L}$=1，且时钟禁止端（第 15 脚）为低电平时，允许时钟输入，这时在时钟脉冲的作用下，数据将由 Q_A 到 Q_H 方向移位。

上图中，TXD(P3.1)作为移位脉冲输出端与所有 74HC165 的移位脉冲输入端 CP 相连；RXD（P3.0）作为串行输入端与 74HC165 的串行输出端 Q_H 相连；P1.0 用来控制 74HC165 的移位与置入而同 S/$\overline{L}$ 相连；74HC165 的时钟禁止端（15 脚）接地，表示允许时钟输入。当扩展多个 8 位输入口时，两芯片的首尾（Q_H 与 S_{IN}）相连。

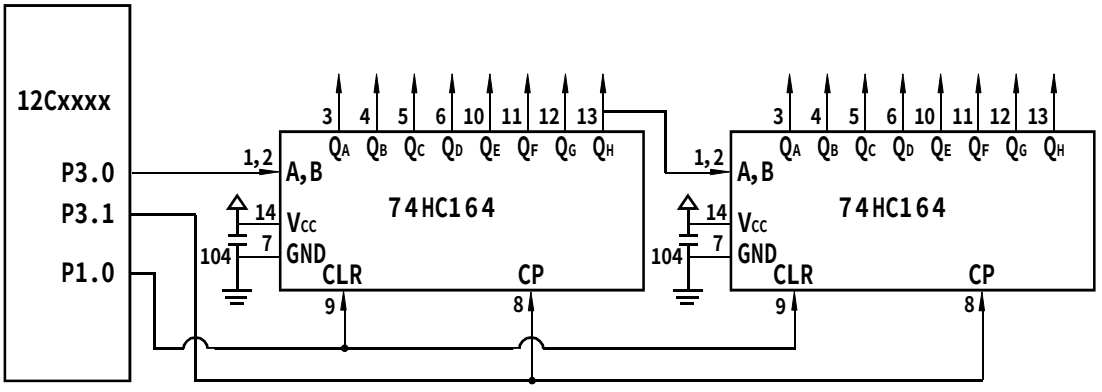
下面的程序是从 16 位扩展口读入 5 组数据（每组二个字节），并把它们转存到内部 RAM 20H 开始的单元中。

```
        MOV   R7，#05H              ；设置读入组数
        MOV   R0，#20H              ；设置内部 RAM 数据区首址
START：  CLR   P1.0                 ；并行置入数据，S/L=0
        SETB  P1.0                  ；允许串行移位 S/L=1
        MOV   R1，#02H              ；设置每组字节数，即外扩 74LS165 的个数
RXDATA： MOV   SCON，#00010000B     ；设串行方式 0，允许接收，启动接收过程
WAIT：   JNB   RI，WAIT             ；未接收完一帧，循环等待
        CLR   RI                    ；清 RI 标志，准备下次接收
        MOV   A，SBUF               ；读入数据
        MOV   @R0，A                ；送至 RAM 缓冲区
        INC   R0                    ；指向下一个地址
        DJNZ  R1，RXDATA          ；为读完一组数据，继续
        DJNZ  R7，START             ；5 组数据未读完重新并行置入
        ……                          ；对数据进行处理
```

上面的程序对串行接收过程采用的是查询等待的控制方式，如有必要，也可改用中断方式。从理论上讲，按上图方法扩展的输入口几乎是无限的，但扩展的越多，口的操作速度也就越慢。

二、用 74HC164 扩展并行输出口

74HC164 是 8 位串入并出移位寄存器。下图是利用 74HC164 扩展二个 8 位输出口的接口电路。



当单片机串行口工作在方式 0 的发送状态时，串行数据由 P3.0（RXD）送出，移位时钟由 P3.1（TXD）送出。在移位时钟的作用下，串行口发送缓冲器的数据一位一位地移入 74HC164 中。需要指出的是，由于 74HC164 无并行输出控制端，因而在串行输入过程中，其输出端的状态会不断变化，故在某些应用场合，在 74HC164 的输出端应加接输出三态门控制，以便保证串行输入结束后再输出数据。

下面是将 RAM 缓冲区 30H、31H 的内容串行口由 74HC164 并行输出的子程序。

```
START：  MOV     R7，#02H          ；设置要发送的字节个数
         MOV     R0，#30H          ；设置地址指针
         MOV     SCON，#00H      ；设置串行口方式 0
SEND：   MOV     A，@R0
         MOV     SBUF，A           ；启动串行口发送过程
WAIT：   JNB     TI，WAIT          ；一帧数据未发送完，循环等待
         CLR     TI
         INC     R0                ；取下一个数
         DJNZ    R7，SEND
         RET
```

附录 C:　　STC11/10xx 系列单片机应用注意事项

关于复位电路：

晶振频率在 12M 以下时:可以不用外部复位电路，原复位电路可以保留，也可以不用，不用时复位脚可经过 1K 电阻短接到地,或者直接短接到地。不过建议设计时 PCB 板上保留 R/C 复位电路，实际使用时再决定用或不 用 。

关于时钟：

如果使用内部 R/C 振荡器时钟(4MHz～8MHz,制造误差加温漂),XTAL1 和 XTAL2 脚浮空.

如果外部时钟频率在27MHz以上时,建议采用实际基本频率就是标称频率的晶体,不要采用三泛音的晶体(基本频率是标称频率的1/3),因为外围参数搭配不当,时钟往往振荡在标称频率的1/3,即基频.或直接使用外部有源石英晶体振荡器,时钟从 XTAL1 脚输入,XTAL2 脚必须浮空.

关于 I/O 口：

少数用户反映 I/O 口有损坏现象,后发现是

有些是 I/O 口由低变高读外部状态时,读不对,实际没有损坏,软件处理一下即可

是因为 1T 的 8051 单片机速度太快了,软件执行由低变高指令后立即读外部状态,此时由于实际输出还没有变高,就有可能读不对,正确的方法是在软件设置由低变高后加 1 到 2 个空操作指令延时,再读就对了.

有些实际没有损坏,加上拉电阻就 OK 了

是因为外围接的是 SPI/I2C 等漏极开漏的电路,要加 10K 上拉电阻.

有些是外围接的是 NPN 三极管,没有加上拉电阻,其实基极串多大电阻,I/O 口就应该上拉多大的电阻,或者将该 I/O 口设置为强推挽输出.

有些确实是损坏了,原因:

发现有些是驱动 LED 发光二极管没有加限流电阻,建议加 1K 以上的限流电阻,至少也要加 470 欧姆以上

发现有些是做行列矩阵按键扫描电路时,实际工作时没有加限流电阻,实际工作时可能出现 2 个 I/O 口均输出为低,并且在按键按下时,短接在一起,我们知道一个CMOS电路的2个输出脚不应该直接短接在一起,按键扫描电路中,此时一个口为了读另外一个口的状态,必须先置高才能读另外一个口的状态,而 8051 单片机的弱上拉口在由 0 变为 1 时,会有 2 个时钟的强推挽高输出电流,输出到另外一个输出为低的 I/O 口,就有可能造成 I/O 口损坏.建议在其中的一侧加 1K 限流电阻,或者在软件处理上,不要出现按键两端的 I/O 口同时为低.

关于电源：

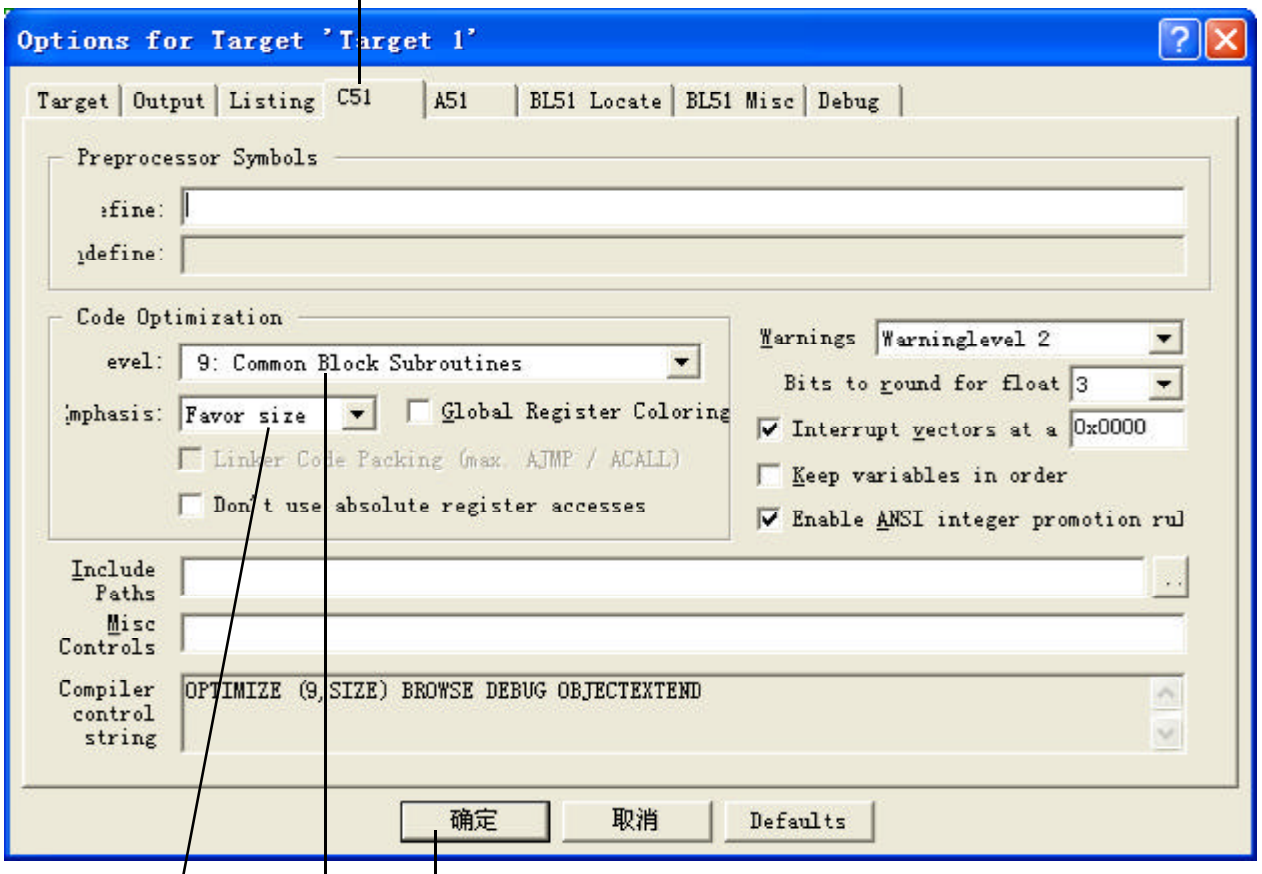
在电源两端应该加一个 47uF 以上的电解电容和一个 0.1uF 的小电容,进行电源去藕滤波.

附录 D:程序空间接近 64K,如何让编译器自动减少代码长度

Keil C51 高级语言编程的软件如何减少代码长度:

1.在“Project”菜单中选择“Options for Target”

2.在“Options for Target”中选择“C51”



3.选择按空间大小， 9 级优化程序

4.重新编译程序即可，如此之后，60K 左右的程序将会下降为 50K 左右

附录E：　　STC11/10xx系列单片机取代传统8051单片机注意事项

STC11/10xx 系列单片机的定时器 0/ 定时器 1/ 串行口与传统 8051 完全兼容，上电复位后，定时器部分缺省还是除 12 再计数的，而串口由定时器 1 控制速度，所以，定时器 / 串口完全兼容。

增加了独立波特率发生器，省去了传统 8052 的定时器 2，如是用 T2 做波特率的，请改用独立波特率发生器做波特率发生器。

传统 8051 的 111 条指令执行速度全面提速，最快的指令快 24 倍，最慢的指令快 3 倍．靠软件延时实现精确延时的程序需要调整。

其它需注意的细节：

ALE：

传统 8051 单片机的 ALE 脚对系统时钟进行 6 分频输出，可对外提供时钟，STC11/10xx 系列不对外输出时钟，如果传统设计利用 ALE 脚对外输出时钟，请利用 STC11/10xx 系列的可编程时钟输出脚对外输出时钟 (CLKOUT0/CLKOUT1/CLKOUT2)或 XTAL2 脚串一个 200 欧姆电阻对外输出时钟．

传统 8051 单片机时钟频率较高时，ALE 脚是一个干扰源，所以 STC89 系列单片机增加了 AUXR 特殊功能寄存器，其中的 Bit0/ALEOFF 位允许禁止 ALE 对系统时钟分频输出。而 STC11/10xx 单片机直接禁止 ALE 脚对系统时钟进行 6 分频输出，彻底清除此干扰源．也有利于系统的抗干扰设计．请自行比较如下的寄存器．

STC89 系列的 AUXR 寄存器：

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset value
AUXR	8Eh	Auxiliary Register 0	-	-	-	-	-	-	EXTRAM	ALEOFF	xxxx,xx00

ALEOFF　0：ALE 脚对系统时钟进行 6 分频输出

1：ALE 脚仅在对外部 64K 数据总线进行 MOVX 指令时才有地址锁存信号输出

STC11/10xx 系列的 AUXR 寄存器：

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
AUXR	8Eh	Auxiliary Register	T0x12	T1x12	UART_M0x6	BRTR	-	BRTx12	XRAM	S1BRS	0000,x000

S1BRS：　0，缺省，串口 1 波特率发生器选择定时器 1，S1BRS 是串口 1 波特率发生器选择位

1，独立波特率发生器作为串口 1 的波特率发生器，此时定时器 1 与串口无关

PSEN：

传统 8031/8032 有 PSEN 信号可以跑外部程序，可以外扩外部程序存储器．现在 STC11/10xx 系列单片机由于是系统晶片概念，内部有大容量程序存储器，不需外扩外部程序存储器，所以直接将PSEN信号去除，可以当普通I/O 口使用．

普通 I/O 口既作为输入又作为输出：

传统8051单片机执行I/O口操作，由高变低或由低变高，以及读外部状态都是12个时钟，而现在STC11/10xx系列单片机执行相应的操作是 4 个时钟．传统 8051 单片机如果对外输出为低，直接读外部状态是读不对的．必须先将 I/O 口置高才能够读对，而传统 8051 单片机由低变高的指令是 12 个时钟，该指令执行完成后，该 I/O 口也确实已变高．故可以紧跟着由低变高的指令后面，直接执行读该 I/O 口状态指令．而 STC11/10xx 系列单片机由于执行由低变高的指令是4个时钟，太快了，相应的指令执行完以后，I/O口还没有变高，要再过一个时钟之后，该 I/O 口才可以变高．故建议此状况下增加 2 个空操作延时指令再读外部口的状态．

P4 口：

最新 STC11/10xx 系列单片机 P4 口地址在 C0H，有完整的 P4 口(P4.0-P4.7)，未扩展外部 INT2/INT3 中断

传统 STC89 系列单片机的 P4 口地址在 E8H，P4 口只有一半(P4.0-P4.3)，P4 有扩展外部 INT2/INT3 中断

如需要 STC11/10 系列单片机的高速性能，又需要在 P4 口上增加 2 个外部中断，请使用 STC12C5Axx 系列单片机

I/O 口驱动能力：

最新 STC11/10xx 系列单片机 I/O 口的灌电流是 20mA，驱动能力超强，驱动大电流时，不容易烧坏．

传统 STC89Cxx 系列单片机 I/O 口的灌电流是 6mA，驱动能力不够强，不能驱动大电流，建议使用 STC11/10xx 系列

中断优先级：

最新 STC11/10xx 系列单片机中断优先级是 2 级，兼容传统 8051

传统 STC89 系列增强型单片机中断优先级是 4 级，增加了 IPH 寄存器，与 IPH 寄存器组合使用，支持 4 级优先级

如需要 STC11/10 系列单片机的高速性能，又需要 4 级中断优先级，请使用 STC12C5Axx 系列单片机

看门狗：

最新 STC11/10xx 系列单片机的看门狗寄存器 WDT_CONTR 的地址在 C1H，增加了看门狗复位标志位

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WDT_CONTR	C1h	Watch-Dog-Timer Control register	WDT_FLAG	-	EN_WDT	CLR_WDT	IDLE_WDT	PS2	PS1	PS0	xx00,0000

传统 STC89 系列增强型单片机看门狗寄存器 WDT_CONTR 的地址在 E1H，没有看门狗复位标志位

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
WDT_CONTR	E1h	Watch-Dog-Timer Control register	-	-	EN_WDT	CLR_WDT	IDLE_WDT	PS2	PS1	PS0	xx00,0000

最新STC11/10xx系列单片机的看门狗在ISP烧录程序可设置上电复位后直接启动看门狗，而传统STC89系列单片机无此功能.故最新 STC11/10xx 系列单片机看门狗更可靠.

EEPROM

STC11/10xx单片机ISP/IAP控制寄存器地址和STC89xx系列单片机ISP/IAP控制寄存器地址不同如下：

Mnemonic	Add	Name	7	6	5	4	3	2	1	0	Reset Value
STC11/10xx系列 IAP_DATA STC89xx 系列 ISP_DATA	C2h E2h	ISP/IAP Flash Data Register									1111,1111
STC11/10xx系列 IAP_ADDRH STC89xx 系列 ISP_ADDRH	C3h E3h	ISP/IAP Flash Address High									0000,0000
STC11/10xx系列 IAP_ADDRL STC89xx 系列 ISP_ADDRL	C4h E4h	ISP/IAP Flash Address Low									0000,0000
STC11/10xx系列 IAP_CMD STC89xx 系列 ISP_CMD	C5h E5h	ISP/IAP Flash Command Register	-	-	-	-	-	-	MS1	MS0	xxxx,xx00
STC11/10xx系列 IAP_TRIG STC89xx 系列 ISP_TRIG	C6h E6h	ISP/IAP Flash Command Trigger									xxxx,xxxx
STC11/10xx系列 IAP_CONTR STC89xx 系列 ISP_CONTR	C7h E7h	ISP/IAP Control Register	IAPEN	SWBS	SWRST	CMD_FAIL -	-	WT2	WT1	WT0	0000,x000

ISP/IAP_TRIG 寄存器有效启动 IAP 操作，需顺序送入的数据不一样：

STC11/10xx 系列单片机的 ISP/IAP 命令要生效，要对 IAP_TRIG 寄存器按顺序先送 5Ah，再送 A5h 方可

STC89xx 系列单片机的 ISP/IAP 命令要生效，要对 IAP_TRIG 寄存器按顺序先送 46h，再送 B9h 方可

EEPROM 起始地址不一样：

STC11/10xx 系列单片机的 EEPROM 起始地址全部从 0000h 开始，每个扇区 512 字节

STC89xx 系列单片机的 EEPROM 起始地址分别有从 1000h/2000h/4000h/8000h 开始的，程序兼容性不够好.

外部时钟和内部时钟：

最新 STC11/10xx 系列单片机有内部 R/C 振荡器作为系统时钟，一般情况下，44/40 脚封装单片机出厂时的设置是使用外部时钟，20/18/16 脚封装单片机出厂时的设置是使用内部 R/C 振荡器作为系统时钟，用户可在 ISP 烧录用户程序时任意选择使用内部 R/C 时钟或外部晶体 / 时钟.

传统 STC89 系列单片机只能使用外部晶体或时钟作为系统时钟.

功耗：

功耗由 2 部分组成，晶体振荡器放大电路的功耗和单片机的数字电路功耗组成，

晶体振荡器放大电路的功耗： 最新 STC11/10xx 系列单片机比 STC89xx 系列低.

单片机的数字电路功耗:时钟频率越高，功耗越大，最新STC11/10xx系列单片机在相同工作频率下，指令执行速度比传统STC89系列单片机快3-24倍，故可用较低的时钟频率工作，这样功耗更低.建议低功耗设计系统外接4-6MHz的晶体或用内部R/C振荡器作为系统时钟，并利用内部的时钟分频器对时钟进行分频，以较低的频率工作，这样单片机的功耗更低

掉电唤醒：

最新 STC11/10xx 系列单片机支持外部中断模式是下降沿就下降沿唤醒，是低电平就低电平唤醒，传统 STC89系列单片机是外部中断口只要是低电平就唤醒，另最新STC11xx系列还有内部专用掉电唤醒定时器可唤醒，另外，STC11xx系列掉电唤醒延时时间可选:32768/16384/8192/4096个时钟，STC89系列固定是1024个时钟