盛群知识产权政策

专利权

盛群半导体公司在全球各地区已核准和申请中之专利权至少有 160 件以上，享有绝对之合法权益。与盛群公司 MCU 或其它产品有关的专利权并未被同意授权使用，任何经由不当手段侵害盛群公司专利权之公司、组织或个人，盛群将采取一切可能的法律行动，遏止侵权者不当的侵权行为，并追讨盛群公司因侵权行为所受之损失、或侵权者所得之不法利益。

商标权

盛群之名称和标识、Holtek 标识、HT-IDE、HT-ICE、Marvel Speech、 Music Micro、 Adlib Micro、 Magic Voice、 Green Dialer、 PagerPro、 Q-Voice、 Turbo Voice、 EasyVoice 和 HandyWriter 都是盛群半导体公司在台湾地区和其它国家的注册商标。

著作权

规格书中所出现的信息在出版当时相信是正确的，然而盛群对于规格内容的使用不负责任。文中提到的应用其目的仅仅是用来做说明，盛群不保证或不表示这些应用没有更深入的修改就能适用，也不推荐它的产品使用在会由于故障或其它原因可能会对人身造成危害的地方。盛群产品不授权使用于救生、维生器件或系统中做为关键器件。盛群拥有不事先通知而修改产品的权利，对于最新的信息，请参考我们的网址 http://www.holtek.com.tw; http://www.holtek.com.cn

技术相关信息

- 工具信息
- FAQs
- 应用范例
  - HT0003S HT48&HT46 MCU 与 HT93LC46 的通信
  - HT0049S HT1380 的读写控制
  - HT0075S MCU 重置电路及振荡电路应用

特性

- 工作电压:
  f_{SYS} = 32768Hz：2.2V～5.5V
  f_{SYS} = 4MHz：2.2V～5.5V
  f_{SYS} = 8MHz：3.3V～5.5V
- 8 个双向输入/输出口
- 最大 16×16 LED 驱动输出
- 8 个 LED 与输入/输出口共用引脚
- 24 个 LED 与输出共用
- 双重边沿触发外部中断输入与输入/输出口共用引脚
- 2 个 8 位可编程定时/计数器和溢出中断
- 外部 RC/XTAL 振荡器和 32768Hz 晶体振荡器
- 双时钟系统提供 3 种工作模式
  —正常模式：RC/XTAL 和 32768Hz 时钟工作
  —慢速模式：只有 32768Hz 时钟工作
  —暂停模式：由看门狗时钟溢出周期性唤醒
- 看门狗定时器
- 4096×15 位的程序存储器
- 192×8 位的数据存储器
- PFD 频率发生器
- HALT 和唤醒功能可降低功耗
- 在 V_{DD}=5V，系统频率为 8MHz 时，指令周期为 0.5μs
- 8 层硬件堆栈
- 8 通道 12 位分辨率 A/D 转换器
- 3 通道 8 位 PWM 输出与输入/输出口共用引脚
- 1/2 偏压 4COM LCD
- 位操作指令
- 查表指令
- 63 条功能强大的指令
- 所有指令执行时间皆为 1 或 2 个指令周期
- 低电压复位功能
- 44/52-pin QFP 封装

概述

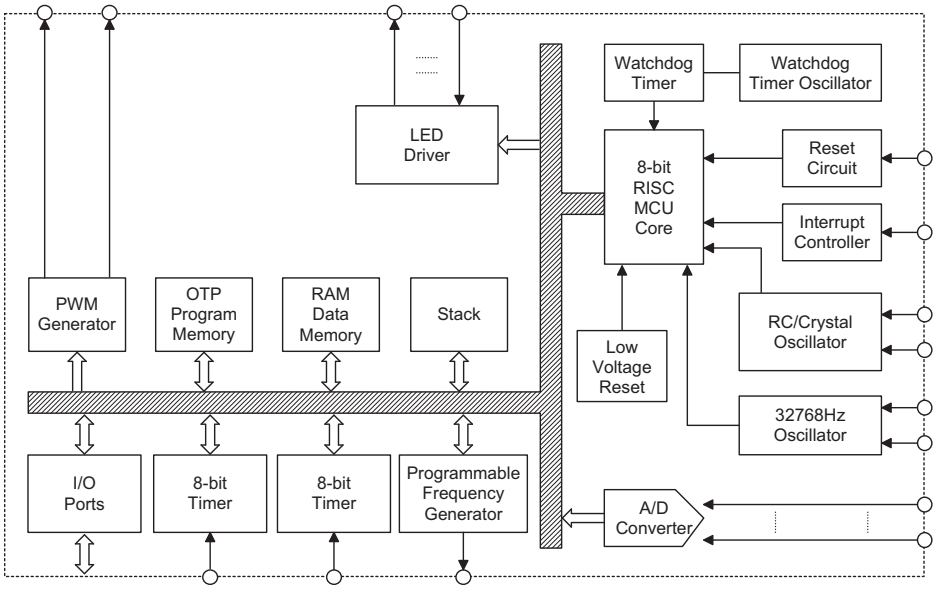
HT46R94 是八位高性能精简指令集单片机，专门为需要 A/D 转换的产品而设计，例如传感器信号输入。同时集成了多通道模数转换器和 3 通道 PWM 输出。当应用到 LED 显示时，简单的接口电路提供大电流驱动。

具有暂停和唤醒功能、振荡类型选择、可编程分频器等功能，使得实际应用时只需要很少的外部器件。

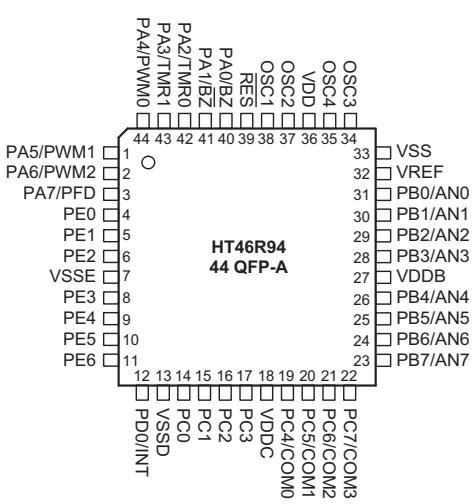
具有 A/D 和 PWM、低功耗、高性能、灵活的输入/输出口和低成本，使得这款单片机可以广泛应用于带传感器信号处理、马达驱动、工业控制、消费类产品和子系统控制器等。

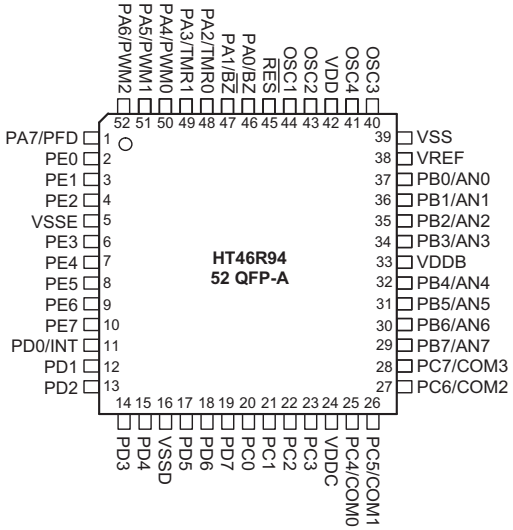
和其他系列单片机一样，HT46R94 由全套专业的硬件和软件工具支持，尽可能地缩短了客户的应用设计和调试周期。

方框图


Watchdog Timer
Watchdog Timer Oscillator
LED Driver
8-bit RISC MCU Core
Reset Circuit
Interrupt Controller
PWM Generator
OTP Program Memory
RAM Data Memory
Stack
Low Voltage Reset
RC/Crystal Oscillator
32768Hz Oscillator
I/O Ports
8-bit Timer
8-bit Timer
Programmable Frequency Generator
A/D Converter

引脚图


HT46R94
44 QFP-A
PA4/PWM0
PA3/TMR1
PA2/TMR0
PA1/BZ
PA0/BZ
RES
OSC1
OSC2
VDD
OSC4
OSC3
PA5/PWM1
PA6/PWM2
PA7/PFD
PE0
PE1
PE2
VSSE
PE3
PE4
PE5
PE6
VSS
VREF
PB0/AN0
PB1/AN1
PB2/AN2
PB3/AN3
VDDB
PB4/AN4
PB5/AN5
PB6/AN6
PB7/AN7
PD0/INT
VSSD
PC0
PC1
PC2
PC3
VDDC
PC4/COM0
PC5/COM1
PC6/COM2
PC7/COM3


HT46R94
52 QFP-A
PA6/PWM2
PA5/PWM1
PA4/PWM0
PA3/TMR1
PA2/TMR0
PA1/BZ
PA0/BZ
RES
OSC1
OSC2
VDD
OSC4
OSC3
PA7/PFD
PE0
PE1
PE2
VSSE
PE3
PE4
PE5
PE6
PE7
PD0/INT
PD1
PD2
VSS
VREF
PB0/AN0
PB1/AN1
PB2/AN2
PB3/AN3
VDDB
PB4/AN4
PB5/AN5
PB6/AN6
PB7/AN7
PC7/COM3
PC6/COM2
PD3
PD4
VSSD
PD5
PD6
PD7
PC0
PC1
PC2
PC3
VDDC
PC4/COM0
PC5/COM1

引脚说明

引脚名称	输入/输出	掩膜选项	说　明
PA0/BZ PA1/$\overline{BZ}$ PA2/TMR0 PA3/TMR1 PA4/PWM0 PA5/PWM1 PA6/PWM2 PA7/PFD	输入/输出	上拉电阻 唤醒 BZ/$\overline{BZ}$ PWM0~PWM2 PFD	8 位双向输入/输出端口，每一位可由掩膜选项设置为唤醒输入。可由软件指令设置为 CMOS 输出或施密特触发输入，可由掩膜选项设置是否带上拉电阻。PA0、PA1、PA2，PA3，PA4，PA5，PA6 和 PA7 分别与 BZ、$\overline{BZ}$、TMR0，TMR1，PWM0，PWM1，PWM2 和 PFD 引脚共用。
PB0 /AN0 PB1 /AN1 PB2 /AN2 PB3 /AN3 PB4 /AN4 PB5 /AN5 PB6 /AN6 PB7 /AN7	输入/输出	—	8 位双向输入/输出口。可由软件指令设置为 CMOS 输出或施密特触发输入。A/D 信号输入与 PB 引脚共用，一旦 PB 作为 A/D 输入（由软件指令设置），则相应输入/输出功能会自动禁止。
PC0~PC3 PC4/COM0 PC5/COM1 PC6/COM2 PC7/COM3	输出	—	8 位 CMOS 输出端口。 PC4~PC7 也可用作 COM0~COM3。
PD0/INT PD1~PD7	输入/输出 输出	—	8 位 CMOS 输出端口。 PD0 和外部中断输入引脚共用。
PE0~PE7	输出	—	8 位 CMOS 输出端口。
OSC1 OSC2	输入 输出	晶体或 RC	OSC1、OSC2 连接 RC 或晶体（由掩膜选项确定）以产生内部系统时钟。在 RC 振荡方式下，OSC2 是系统时钟四分频的输出口。
OSC3 OSC4	输入 输出	—	OSC3 和 OSC4 连接到一个 32768Hz 的晶体振荡器来产生系统时钟和实时系统时钟。
$\overline{RES}$	输入	—	施密特触发复位输入端，低电平有效。
VDD	—	—	正电源。
VSS	—	—	负电源，接地。
VREF			A/D 基准电压输入端。
VDDB VDDC	—	—	PB&PC 端口正电源输入。
VSSD VSSE	—	—	PD&PE 端口负电源输入，接地。

注：1、PA 端口每位可通过掩膜选项设置为唤醒输入。
　　2、表格中提供的是芯片最大封装尺寸时的引脚，在小的封装中并非都存在。

极限参数

电源供应电压　……　V_{SS}－0.3V~ V_{SS}＋6.0V
端口输入电压　…… V_{SS}－0.3V~ V_{DD}＋0.3V
I_{OL} 总电流…… ……. …….150mA
总消耗电流…… ……. …….500mW
储存温度　……　－50℃~ 125℃
工作温度　……　－40℃~ 85℃
I_{OH} 总电流…… …. -100mA

注：　这里只强调额定功率，超过极限参数所规定的范围将对芯片造成损害，无法预期芯片在上述标示范围外的工作状态，而且，若长期在标示范围外的条件下工作，可能影响芯片的可靠性。

直流电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压	—	f_{SYS}=32768Hz	2.2	—	5.5	V
		—	f_{SYS}=4MHz	2.2	—	5.5	
		—	f_{SYS}=8MHz	3.3	—	5.5	
V_{AVDD}	模拟电路工作电压	—	V_{AVDD}= V_{DD}	2.7	—	5.5	V
I_{DD1}	工作电流(晶体，RC 振荡)	3V	无负载 f_{SYS}=4MHz ADC 禁止	—	1	2	mA
		5V		—	2.5	5	
I_{DD2}	工作电流(RC，RTC 振荡)	5V	无负载 f_{SYS}=8MHz ADC 禁止	—	4	8	mA
I_{DD3}	工作电流(RTC*振荡，RC 振荡关闭)	3V	无负载 f_{SYS}=32768Hz ADC 禁止	—	20	40	μA
		5V		—	50	100	
I_{STB1}	静态电流(看门狗&RTC*振荡打开)	3V	无负载 系统暂停模式	—	3	5	μA
		5V		—	6	10	
I_{STB2}	静态电流(看门狗关闭，RTC*振荡打开)	3V	无负载 系统暂停模式	—	1	2	μA
		5V			2	4	
I_{STB3}	静态电流(看门狗打开，RTC*振荡关闭)	3V	无负载 系统暂停模式	—	2	4	μA
		5V		—	4	8	
I_{STB4}	静态电流(看门狗&RTC*振荡关闭)	3V	无负载 系统暂停模式	—	—	1	μA
		5V		—	—	2	
I_{STB5}	200K 时静态电流（见注 2）1/2VDD 偏压电阻（看门狗&RTC 振荡关闭）	3V	无负载 系统暂停模式	—	30	50	μA
V_{IL1}	输入/输出口的低电平输入	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	输入/输出口的高电平输入	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	低电平输入电压（$\overline{RES}$）	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	高电平输入电压（$\overline{RES}$）	—	—	$0.9V_{DD}$	—	V_{DD}	V
V_{LVR}	低电压复位	—	—	1.98	2.1	2.22	V
				2.98	3.15	3.32	
				3.98	4.2	4.42	
I_{OL1}	PA 输入/输出口灌电流	3V	V_{OL}=0.1V_{DD}	4	8	—	mA
		5V		10	20	—	
I_{OL2}	PB、PC 输入/输出口灌电流	3V	V_{OL}=0.1V_{DD}	0.6	1.3	—	mA
		5V		1.5	3.0	—	
I_{OL3}	PD、PE 输入/输出口灌电流	3V	V_{OL}=0.1V_{DD}	12	24	—	mA
		5V		30	60	—	
I_{OH1}	PA 输入/输出口源电流	3V	V_{OH}=0.9V_{DD}	-2	-4	—	mA
		5V		-5	-10	—	
I_{OH2}	PB、PC 输入/输出口源电流	3V	V_{OH}=0.9V_{DD}	-4	-8	—	mA
		5V		-10	-20	—	
I_{OH3}	PD、PE 输入/输出口源电流	3V	V_{OH}=0.9V_{DD}	0.25	0.5	—	mA
		5V		0.5	1.0	—	
R_{PH}	上拉电阻	3V	—	20	60	100	kΩ
		5V		10	30	50	

V_{BIAS}	1/2V_{DD} 偏压（见注 3）	5V	—	2.3	2.5	2.7	V
V_{AD}	A/D 输入电压	—	—	0	—	V_{REF}	V
V_{REF}	A/D 输入基准电压范围	—	V_{AVDD} =2.7V~5.5V	1.6	—	V_{AVDD} +0.1	V
DNL	A/D 微分非线性度	—	V_{AVDD}=5V V_{REF}= V_{AVDD} t_{AD}=0.5μS	-2	—	2	LSB
INL	A/D 积分非线性度			-4	—	4	
I_{ADC}	A/D 转换时增加的功耗	3V	—	—	0.5	1.0	mA
		5V		—	1.5	3.0	

注：1、*RTC 振荡是慢起振振荡器

2、将 LCDC 寄存器（1FH）中 LCDEN=1，COM0EN=1 和 RSEL=0 测试 I_{STB5}

3、V_{BIAS} 电压为设计时保证，并非测试值

4、V_{AVDD} 提供内部模拟电路电压，芯片内部与 V_{DD} 内部连接，没有外部引脚

交流电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
f_{SYS1}	系统时钟（RC 振荡）	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	
f_{SYS2}	系统时钟（RTC 振荡）	—	2.2V~5.5V	—	32768	—	kHz
f_{TIMER}	定时器的输入频率	—	2.2V~5.5V	0	—	4000	kHz
		—	3.3V~5.5V	0	—	8000	
t_{WDTOSC}	看门狗振荡器周期	3V	—	45	90	180	μs
		5V	—	32	65	130	
t_{FSP1}	f_{SP} 溢出周期 (时钟源为 WDT 振荡)	3V	预分频（fs/4096）	184	369	737	ms
		5V		131	266	532	
t_{FSP2}	f_{SP} 溢出周期 (时钟源为 RTC 振荡)	—	预分频（fs/4096）	—	125	—	ms
t_{RES}	外部复位低电平脉冲宽度	—	—	1	—	—	μs
t_{SST}	系统启动延时周期	—	暂停模式唤醒	—	1024	—	*t_{SYS}
t_{LVR}	低电压复位	—	—	0.25	1	2	ms
t_{INT}	中断脉冲宽度	—	—	1	—	—	μs
t_{AD}	A/D 时钟周期	—	—	0.5	—	—	μs
t_{ADC}	A/D 转换时间	—	—	—	16	—	t_{AD}

注：*t_{SYS} =1/ f_{SYS1} 或 1/ f_{SYS2}

系统结构

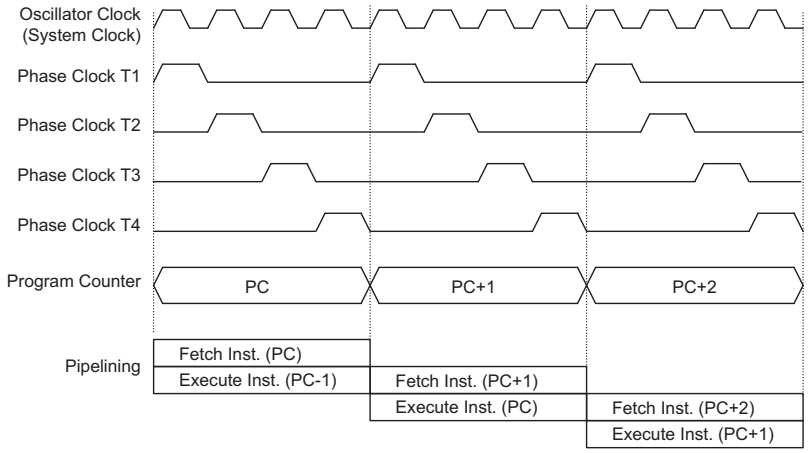
内部系统结构是盛群半导体公司经济型 A/D 单片机具有良好运行性能的主要因素。由于采用 RISC 结构，此系列单片机具有高运算速度和高性能的特性。通过流水线的方式，指令的取得和执行同时进行，此举使得除了分支和调用指令外，其它指令都能在一个指令周期内完成。8 位的 ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、加、减和分支等功能，而内部的数据路径则以通过累加器或 ALU 的方式加以简化。有些寄存器在数据存储器中被实现，且可以直接或间接寻址。简单的寄存器寻址方式和结构特性，确保了在提供最大可靠度和灵活性的 I/O 和 A/D 控制系统时，仅需要少数的外部器件。

时序和流水线结构

系统时钟由晶体/陶瓷振荡器或 RC 振荡器提供，细分为 T1~T4 四个内部产生的非重叠时钟。程序计数器在 T1 时自动加一并抓取一条新的指令，剩下的 T2~T4 时钟完成指令的解码和执行，因此一个 T1~T4 时钟形成一个指令周期。虽然指令的取得和执行发生在连续的指令周期，但单片机流水线的结构会保证指令在一个指令周期内被有效的执行，特殊的情况发生在程序计数器的内容被改变的时候，如子程序的调用或跳转，在这情况下指令将需要多一个指令周期的时间去执行。

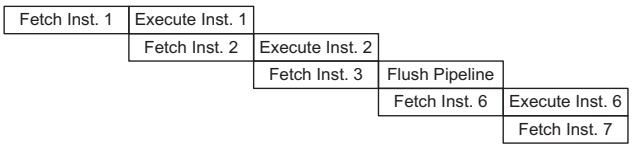
当使用 RC 振荡器时，OSC2 可以获得一个 T1 相时钟同步信号，这个 T1 相时钟有 $f_{SYS}/4$ 的频率，拥有 1:3 高/低的占空比。

如果指令牵涉到分支，例如跳转或调用等指令，则需要两个机器周期才能完成指令执行。需要一个额外周期的原因是程序先用一个周期取出实际要跳转或调用的地址，再用另一个周期去实际执行分支动作，因此用户必须特别考虑额外周期的问题，尤其是在执行时间要求较严格时。



系统时序和流水线

```
1          MOV A,[12H]
2          CALL DELAY
3          CPL [12H]
4          :
5          :
6  DELAY:  NOP
```



指令捕捉

程序计数器

程序执行期间，程序计数器用来指向下一条要执行的指令地址。除了 JMP 或 CALL 这些要求跳转到一个非连续的程序存储器地址之外，它会在每条指令执行完后自动增加一。然而必须要注意只有低 8 位，即程序计数器低字节寄存器 PCL，是可以让使用者直接读写的。

当执行的指令要求跳转到非连续的地址时，如跳转指令、子程序调用、中断或复位等，单片机通过载入所需的地址到程序计数器来控制程序。对于条件跳转指令，一旦条件符合，下一条在现在指令执行时所取得的指令即会被摒弃，而由一个空指令周期来加以取代。

程序计数器低字节，即程序计数器低字节寄存器 PCL，可以通过程序控制取得，且它是可以读取和写入的寄存器。通过直接传送数据到这寄存器，一个程序短跳转可以直接被执行，然而因为只有低字节的运用是有效的，因此跳转被限制在同页存储器，即 256 个存储器地址的范围内，当这样一个程序跳转要执行时，需注意会插入一个空指令周期。

程序计数器低字节在程序控制下是完全可用的。PCL 的使用可能导致程序分支，所以额外的周期需要预先取得。有关 PCL 寄存器更多的信息可在特殊功能寄存器部份中找到。

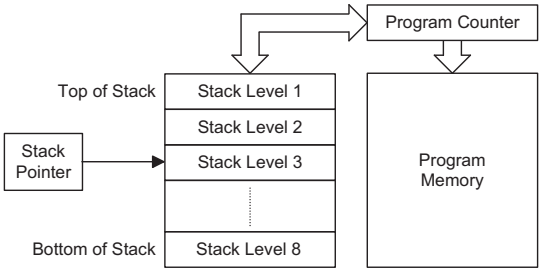
模式	程序计数器											
	b11	b10	b9	b 8	b 7	b 6	b 5	b 4	b 3	b 2	b 1	b 0
初始化复位	0	0	0	0	0	0	0	0	0	0	0	0
外部中断	0	0	0	0	0	0	0	0	0	1	0	0
定时/计数器 0 溢出中断	0	0	0	0	0	0	0	0	1	0	0	0
定时/计数器 1 溢出中断	0	0	0	0	0	0	0	0	1	1	0	0
时基中断	0	0	0	0	0	0	0	1	0	0	0	0
A/D 转换中断	0	0	0	0	0	0	0	1	0	1	0	0
装载 PCL	PC11	PC10	PC 9	PC 8	@7	@6	@5	@4	@3	@2	@1	@0
条件跳跃	PC+2											
跳转、子程序调用	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
子程序返回	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

程序计数器

注： PC11~PC8：程序计数器当前位
@7~@0：PCL 字节
#11~#0：指令地址位
S11~S0：堆栈指针
程序计数器 12 位宽，即 b11~b0

堆栈

堆栈是存储器中一个特殊的部分，它只用来储存程序计数器中的内容。HT46R94 有 8 层堆栈，它既不是数据部份也不是程序空间部份，且既不可读取也不可写入。当前层由堆栈指针 SP 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入堆栈。当子程序或中断服务程序结束时，返回指令（RET 或 RETI）使程序计数器从堆栈中返回它之前的值。当芯片复位之后，SP 将指向堆栈的顶部。



如果堆栈已满，且有非屏蔽的中断发生，中断请求标志位会被置位，但是中断响应将被禁止。当堆栈指针减少（执行 RET 或 RETI），中断将被响应。这个特性提供程序设计者简单的方法来预防堆栈溢出。然而即使堆栈已满，CALL 指令仍然可以被执行，但会造成堆栈溢出。使用时应避免堆栈溢出的情况发生，因为这可能会造成不可预期的程序分支指令执行错误。

算术逻辑单元

算术逻辑单元是单片机中很重要的部份，执行指令集中的算术和逻辑运算。ALU 连接到单片机的数据总线，在接收相关的指令码后执行需要的算术与逻辑运算，并将结果储存在指定的寄存器，当 ALU 计算或操作时，可能导致进位、借位或其它状态的改变，而相关的状态寄存器会因此更新内容以显示这些改变，ALU 所提供的功能如下：

- 算术运算：ADD、ADDM、ADC、ADCM、SUB、SUBM、SBC、SBCM、DAA
- 逻辑运算：AND、OR、XOR、ANDM、ORM、XORM、CPL、CPLA
- 移位运算：RRA、RR、RRCA、RRC、RLA、RL、RLCA、RLC
- 递增和递减：INCA、INC、DECA、DEC
- 分支判断：JMP、SZ、SZA、SNZ、SIZ、SDZ、SIZA、SDZA、CALL、RET、RETI

程序存储器

程序存储器用来存放用户代码即储存程序。HT46R94 程序存储器是一次可编程（OTP），使用者可编写他们的应用代码到单片机中。使用适当的编程工具，OTP 单片机可以提供使用者灵活的方式来自由开发他们的应用，这对于除错或需要经常升级与改变程序的产品是很有帮助的。对于中小型量产，OTP 亦为极佳的选择。

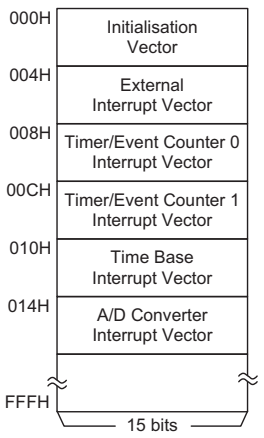
结构

15 位的程序存储器的容量是 4K。程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口，数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。

特殊向量

程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。

- 地址 000H
  该地址为程序初始化保留。系统复位后，程序总是从 000H 开始执行。
- 地址 004H
  该地址为外部中断服务程序保留。当外部中断引脚收到上升或下降沿跳变信号，如果中断允许且堆栈未满，则程序会跳转到 004H 地址开始执行。
- 地址 008H
  该地址为定时/计数器 0 中断服务程序保留。当定时/计数器 0 溢出，如果中断允许且堆栈未满，则程序会跳转到 008H 地址开始执行。
- 地址 00CH
  该地址为定时/计数器 1 中断服务程序保留。当定时/计数器 1 溢出，如果中断允许且堆栈未满，则程序会跳转到 00CH 地址开始执行。
- 地址 010H
  该地址为时基中断服务程序保留。当时基中断发生，如果中断允许且堆栈未满，则程序会跳转到 010 H 地址开始执行。
- 地址 014H
  该地址为 A/D 转换中断服务程序保留。当 A/D 转换完成，如果中断允许且堆栈未满，则程序会跳转到 014H 地址开始执行。

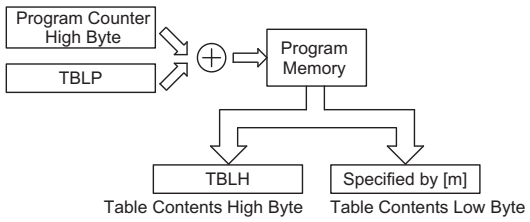


查表

程序存储器中的任何地址都可以定义成一个表格，以便储存固定的数据。使用表格时，表格指针必须先行设定，其方式是将表格的低字节地址放在表格指针寄存器 TBLP 中。这个寄存器定义表格低字节的 8 位地址。

在设定完表格指针后，表格数据可以使用“TABRDC [m]”或“TABRDL [m]”指令从当前的程序所在的存储器页或存储器最后一页中来查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到使用者所指定的数据存储器[m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊寄存器，而高字节中未使用的位将被读取为“0”。

下图是查表中寻址/数据流程：



查表范例

以下范例说明 HT46R94 中，表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器的最后一页，在此 ORG 伪指令中的值为 F00H，即 4K 程序存储器最后一页存储器的起始地址，而表格指针的初始值则为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址 F06H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRDC [m]”指令被使用，则表格指针指向当前页。在这个例子中，表格数据的高字节等于零，而当“TABRDL [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

```
tempreg1   db ? ;temporary register #1
tempreg2   db ? ;temporary register #2
     :
     :

mov    a,06h    ;initialize table pointer - note that this address
                ;is referenced

mov    tblp,a   ;to the last page or present page
     :
     :

tabrdl tempreg1 ;transfers value in table referenced by table pointer
                ;to tempreg1 data at prog. memory address F06H transferred to
                ;tempreg1 and TBLH

dec  tblp       ;reduce value of table pointer by one

tabrdl tempreg2 ;transfers value in table referenced by table pointer
                ;to tempreg2 data at prog. memory address F05H transferred to
                ;tempreg2 and TBLH in this example the data "1A" is transferred
                ;to tempreg1 and data  "0F" to register tempreg2
                ;the value "0" will be transferred to the high byte register TBLH
     :
     :

org  F00h       ;sets initial address of last page

dc   00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
     :
```

TBLH 寄存器为只读寄存器，不能重新储存，若主程序和中断服务程序都使用表格读取指令，应该注意它的保护。使用表格读取指令，中断服务程序可能会改变 TBLH 的值，若随后在主程序中再次使用这个值，则会发生错误，因此建议避免同时使用表格读取指令。然而在某些情况下，如果同时使用表格读取指令是不可避免的，则在执行任何主程序的表格读取指令前，中断应该先除能，另外要注意的是所有与表格相关的指令，都需要两个指令周期去完成操作。

指令	表格地址											
	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TABRDC [m]	PC11	PC10	PC9	PC8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

表格区

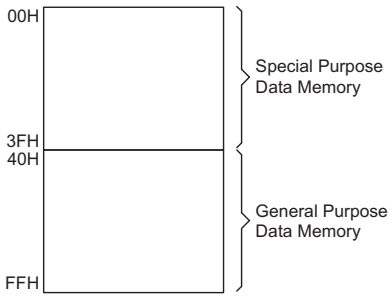
注：PC11~PC8：程序指针当前位
@7~@0：表格指针 TBLP 位
表格地址为 12 位，即 b11~b0

数据存储器

数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。分为两部份，第一部份是特殊功能寄存器，这些寄存器有固定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，但有些被加以保护而不对用户开放。第二部份数据存储器是作一般用途使用，都可在程序控制下进行读取和写入。

结构

数据存储器的两个部份，即特殊和通用数据存储器，位于连续的地址。全部 RAM 为 8 位宽度，但存储器长度因所选择的单片机而不同。所有单片机的数据存储器的起始地址都是 00H。常见的寄存器，如 ACC 和 PCL 等，全都具有相同的数据存储器地址。



数据存储器结构

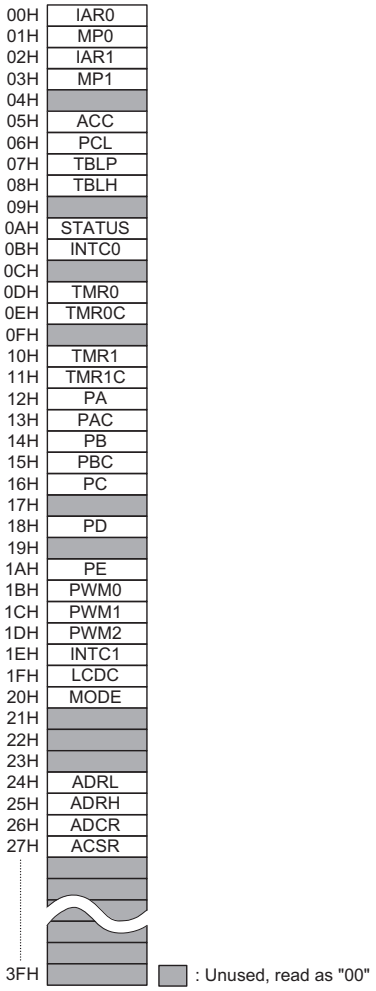
注意：除部分特殊位，大部分数据存储区可以通过“SET [m].i”和“CLR [m].i”直接寻址。数据存储器也可以通过指针 MP0 和 MP1 间接寻址。

通用数据存储器

所有的单片机程序需要一个读/写的存储区，让临时数据可以被储存和再使用，该 RAM 区域就是通用数据存储器。这个数据存储区可让使用者进行读取和写入的操作。使用“SET [m].i”和“CLR [m].i”指令可对个别的位做置位或复位的操作，方便用户在数据存储器内进行位操作。

特殊数据存储器

这个区域的数据存储器是存放特殊寄存器的，这些寄存器与单片机的正确操作密切相关，大多数的寄存器可进行读取和写入，只有一些是被保护而只能读取的，相关细节的介绍请参看有关特殊功能寄存器的部份。要注意的是，任何读取指令对存储器中未定义的地址进行读取将得到“00H”的值。



特殊数据存储器

特殊功能寄存器

为了确保单片机能成功的工作，数据存储器中设置了一些内部寄存器。这些寄存器确保内部功能（如定时器和中断等）和外部功能（如 I/O 数据控制和 A/D 转换操作）的正确工作。在数据存储器中，这些寄存器以 00H 作为起始地址。在特殊功能寄存器和通用数据存储器的起始地址之间，有一些未定义的数据存储器，被保留用来做未来扩充，若从这些地址读取数据将返回 00H 值。

间接寻址寄存器 — IAR0，IAR1

IAR0 和 IAR1 寄存器位于数据存储区的 00H 和 02H 地址，并没有实际的物理地址。间接寻址的方法准许使用间接寻址指针做数据操作，以取代定义实际存储器地址的直接存储器寻址方法。在间接寻址寄存器上的任何动作，将对间接寻址指针(MP0 和 MP1)所指定的存储器地址产生对应的读/写操作。直接读取 IAR0 和 IAR1 寄存器将返回 00H 的结果，而直接写入此寄存器则不做任何操作。

间接寻址指针 — MP0，MP1

间接寻址指针 MP0 和 MP1 位于数据存储器中，能象普通的寄存器一般被写入和操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由间接寻址指针所指定的地址。

以下例子说明如何清除一个具有 4 RAM 地址的区块，它们已事先定义成地址 adres1 到 adres4。

```
data .section ´data´
adres1     db ?
adres2     db ?
adres3     db ?
adres4     db ?
block      db ?
code .section at 0 ´code´
org  00h
start:
     mov a,04h           ; setup size of block
     mov block,a
     mov a,offset adres1 ; Accumulator loaded with first RAM address
     mov mp0,a           ; setup memory pointer with first RAM address

loop:
     clr IAR0            ; clear the data at address defined by mp0
     inc mp0             ; increment memory pointer
     sdz block           ; check if last memory location has been cleared
     jmp loop

continue:
```

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

累加器 — ACC

对任何单片机来说，累加器是相当重要的且与 ALU 所完成的运算有密切关系，所有 ALU 得到的运算结果都会暂时储存在 ACC 累加器。若没有累加器，ALU 必须在每次进行如加法、减法和移位的运算时，将结果写入到数据存储器，这样会造成程序编写和时间的负担。另外数据传送也常常牵涉到累加器的临时储存功能，例如在使用者定义的寄存器和另一个寄存器之间传送数据时，由于两寄存器之间不能直接传送数据，因此必须通过累加器来传送数据。

程序计数器低字节寄存器 — PCL

为了提供额外的程序控制功能，程序计数器低字节设置在数据存储器的特殊功能区域内，程序员可对此寄存器进行操作，很容易的直接跳转到其它程序地址。直接给 PCL 寄存器赋值将导致程序直接跳转到程序存储器的某一地址，然而由于寄存器只有 8 位宽度，因此只允许在本页的程序存储器范围内进行跳转，而当使用这种运算时，要注意会插入一个空指令周期。

表格寄存器 — TBLP，TBLH

这两个特殊功能寄存器对储存在程序存储器中的表格进行操作。TBLP 为表格指针，指向表格数据的地址。它的值必须在任何表格读取指令执行前加以设定，由于它的值可以被如 INC 或 DEC 的指令所改变，这就提供了一种简单的方法对表格数据进行读取。表格读取数据指令执行之后，表格数据高字节存储在 TBLH 中。其中要注意的是，表格数据低字节会被传送到使用者指定的地址。

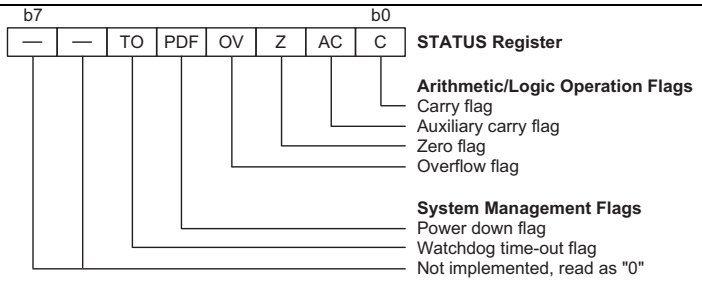
状态寄存器 — STATUS

状态寄存器由零标志位（Z），进位标志位（C），辅助进位标志位（AC），溢出标志位（OV），暂停标志位（PDF），看门狗定时器溢出标志位（TO）组成。该寄存器不仅记录状态信息，还控制运算流程。

除了 TO 和 PDF 以外，状态寄存器中的位都与其它寄存器一样，可用指令来改变。任何写到状态寄存器的数据不会改变 TO 或 PDF 标志位。但是与状态寄存器有关的操作会导致状态寄存器的改变。系统上电，看门狗定时器溢出，或是执行 HALT 指令，都会影响 TO 标志位。改变 PDF 标志位的操作，包括系统上电，执行 HALT 指令和执行 CLR WDT 指令。

Z、OV、AC 和 C 标志位通常反映最近运算的状态：

- 当加法运算的结果产生进位，或减法运算的结果没有产生借位时，则 C 被置位，否则 C 被清零，同时 C 也会被带进位/借位的移位指令所影响。
- 当低半字节加法运算的结果产生进位，或高半字节减法运算的结果没有产生借位时，AC 被置位，否则 AC 被清零。
- 当算术或逻辑运算结果是零时，Z 被置位，否则 Z 被清零。
- 当运算结果高两位的进位状态异或结果为 1 时，OV 被置位，否则 OV 被清零。
- 系统上电或执行“CLR WDT”指令会清零 PDF，而执行“HALT”指令则会置位 PDF。
- 系统上电或执行“CLR WDT”或“HALT”指令会清零 TO，而当 WDT 溢出则会置位 TO。



状态寄存器

另外当进入一个中断程序或执行子程序调用时，状态寄存器不会自动压入到堆栈保存。假如状态寄存器的内容是重要的且子程序可能改变状态寄存器的话，则需谨慎的去做正确的储存。

中断控制寄存器 — INTC0，INTC1

8 位的 INTC0 和 INTC1 寄存器用来控制外部和内部中断的动作。通过标准的位操作指令来设定这些寄存器的位，每个中断的使能/除能功能可分别被控制。寄存器内的总中断位 EMI 控制所有中断的使能/除能，用来设定所有中断使能位的开或关。当一个中断程序被响应时，就会自动屏蔽其它中断，EMI 位将被清除，而执行“RETI”指令则会置位 EMI 位。

定时/计数器 — TMR0，TMR0C，TMR1，TMR1C

HT46R94 提供了 2 个 8 位定时/计数器。TMR0 和 TMR1 寄存器用于存放 8 位设定值，可以预先写入固定的数据，以允许设定不同的时间中断。而 TMR0C 和 TMR1C 寄存器设置定时/计数器的工作模式、打开或关闭定时/计数器。

输入/输出控制寄存器

在特殊功能寄存器中，输入/输出寄存器和它们相对应的控制寄存器很重要。所有输入/输出和输出端口都有相对应的寄存器，标示为 PA、PB、PC，PD 和 PE。如数据存储器结构图中所示，这些输入/输出寄存器映射到数据存储器的特定地址，用以传送端口上的输入/输出数据。输入/输出端口 PA 和 PB 对应的控制寄存器，分别为 PAC 和 PBC，也同样映射到数据存储器的特定地址。这些控制寄存器设定引脚的状态，以决定哪些是输入口，哪些是输出口。要设定一个引脚为输入，控制寄存器对应的位必须设定成逻辑高，若引脚设定为输出，则控制寄存器对应的位必须设为逻辑低。程序初始化期间，在从输入/输出端口中读取或写入数据之前，必须先设定控制寄存器的位以确定引脚为输入或输出。使用“SET [m].i”和“CLR [m].i”指令可以直接设定这些寄存器的某一位。这种在程序中可以通过改变输入/输出端口控制寄存器中某一位而直接改变该端口输入/输出状态的能力是此系列单片机非常有用的特性。

PC，PD 和 PE 端口仅作为输出使用，因此没有相应的控制寄存器。当输出寄存器中位信息设置为高时，对应 NMOS 输出为高组态；当输出寄存器中位信息设置为低时，输出为低电平状态。

脉宽调制寄存器 — PWM0，PWM1，PWM2

HT46R94 提供了 3 个脉冲宽度调制器(即 PWM0，PWM1 和 PWM2)。每个 PWM 都具有自己独立的控制寄存器。这些 8 位的寄存器定义相应的脉冲宽度调制器调制周期的占空比。

A/D 转换寄存器 — ADRL，ADRH，ADCR，ACSR

HT46R94 提供了 8 通道 12 位 A/D 转换器。A/D 转换需要涉及到 2 个数据存储器、1 个控制寄存器和 1 个时钟控制寄存器。A/D 转换结果寄存器为高位 ADRH 寄存器和低位 ADRL 寄存器。当一个模数转换周期结束后，转换出的数字量将保存在这些寄存器中。通道的选择和 A/D 转换器的设置通过寄存器 ADCR 控制，A/D 时钟频率由时钟源寄存器 ACSR 定义。

模式寄存器 — MODE

MODE 寄存器用来选择系统工作在正常模式，低速模式或暂停模式。模式寄存器也包含 1 位控制 32768Hz 晶体振荡器的快速启动功能。

LCD 控制寄存器 — LCDC

LCDC 寄存器用来设置 LCD 显示的各种功能，比如 COM 端口的 1/2 偏压使能设置，偏压电阻和 LCD 使能。

输入/输出端口

盛群单片机的输入/输出端口控制具有很大的灵活性。这体现在 PA，PB 每一个引脚在使用者的程序控制下可以被指定为输入或输出、所有引脚的上拉电阻选项、以及指定引脚的唤醒选择，这些特性也使得此类单片机在广泛应用上都能符合开发的要求。

HT46R94 提供最多 16 位的双向输入/输出口 PA 和 PB，以及输出口 PC，PD 和 PE。这些输入/输出端口在数据存储器的对应指定地址如表所示。作为输入操作时，输入/输出引脚无锁存功能，输入数据必须在指令“MOV A,[m]”T2 上升沿准备好，m 表示端口地址。对于输出操作，所有数据是锁存的，而且持续到输出锁存被重写。

上拉电阻

许多产品应用在端口处于输入状态时需要外加一个上拉电阻来实现上拉的功能。为了免去这个外加的电阻，当引脚设置为输入时，可由内部连接上拉电阻，这些上拉电阻可通过掩膜选项来加以选择，它用一个 PMOS 晶体管来实现。

PA 口唤醒

此系列的单片机都具有暂停功能，使得单片机进入暂停模式以降低功耗，此功能对于电池及低功耗应用很重要。唤醒单片机有很多种方法，其中之一就是使 PA 某位引脚从高电平转为低电平。当使用暂停指令“HALT”迫使单片机进入暂停状态以后，单片机将保持闲置即低功耗状态，直到 PA 口上被选为唤醒输入的引脚电平发生下降沿跳变。这个功能特别适合于通过外部开关来唤醒的应用。值得注意的是 PA 口的每个引脚都可单独的选择具有唤醒的功能。

输入/输出端口控制寄存器

PA 和 PB 为输入/输出端口，具有各自的控制寄存器 PAC 和 PBC 控制输入/输出状态。利用此控制寄存器，每一个 CMOS 输出或者输入不管有没有上拉电阻设置，均可利用软件控制方式加以动态的重新设置。所有输入/输出端口的引脚都各自对应于输入/输出端口控制寄存器的某一位。若输入/输出引脚要实现输入功能，则对应的控制寄存器位必须设定为“1”，这时程序指令可以直接读出输入引脚的逻辑状态。如果引脚的控制寄存器位被设定为“0”，则此引脚被设置为 CMOS 输出。当引脚被设置为输出状态，程序指令读取的是输出端口寄存器的内容。请注意当输入/输出端口被设置为输出状态时，此时如果对输出口做读取的动作，则会读取到内部数据寄存器中的锁存值，而不是输出引脚实际的逻辑状态。

引脚共用功能

引脚的共用功能可以增加单片机的灵活程度。有限的引脚个数会严重的限制设计者，但是引脚的复用功能特性，可以很好的解决此类问题。复用功能输入/输出引脚的功能选择，有些是由掩膜选项进行设定，有些则是在应用程序中进行控制。

- 外部中断输入
  外部中断引脚 INT 为双重边沿触发，与输出引脚 PD0 共用。如果需要外部中断输入，INTC0 控制寄存器中的外部中断使能位必须设置为 1，且 LCDC 寄存器中 LCDEN 位设置为 0，同时将 PD0 设置为高电平。如果外部中断使能位没有设置为 1，PD0 将作为 CMOS 输出引脚使用。

- 外部时钟输入
  外部时钟输入引脚 TMR0 和 TMR1 分别与 PA2 和 PA3 共用。如果引脚被设定为计数器输入，则定时/计数控制寄存器中相应的控制位也必须正确设置。在不需要外部计数器输入的时候，此引脚也可以作为一般 I/O 引脚使用。对于此种应用，TMRC 控制寄存器中的定时器模式位必须选为定时器模式(内部时钟源)，以避免输入/输出引脚与计数器操作的冲突。

- 蜂鸣器输出
  HT46R94 提供蜂鸣器功能，引脚与 PA7 共用。蜂鸣器输出功能通过掩膜选项进行设置，并且在程序烧录后保持不变。需要注意的是，必须将 PAC 中相应的位 PAC.7 设为输出以使能 PFD 输出。若 PAC 设置为带上拉电阻的输入，即使选择了 PFD 输出功能，此引脚仍将作为带上拉电阻的一般输入引脚使用。

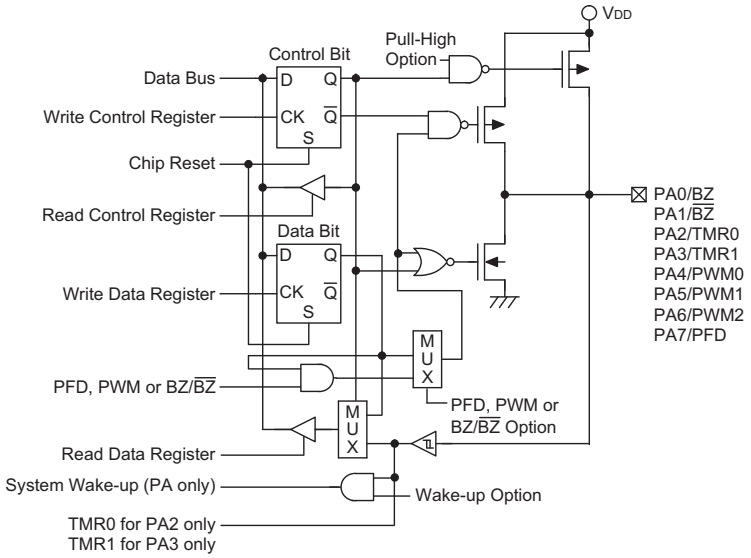
- PWM 输出
  HT46R94 提供 3 个 PWM 输出，即 PWM0，PWM1 和 PWM2，分别与 PA4 ，PA5 和 PA6 共用引脚。PWM 输出功能可以通过掩膜选项进行选择，并且在程序烧录后保持不变。需要注意的是，必须将 PAC 中相应的位设为输出以使能 PWM 输出。若 PAC 设置为带上拉电阻的输入，即使选择了 PWM 输出功能，此引脚仍将作为带上拉电阻的一般输入引脚使用。

- A/D 输入

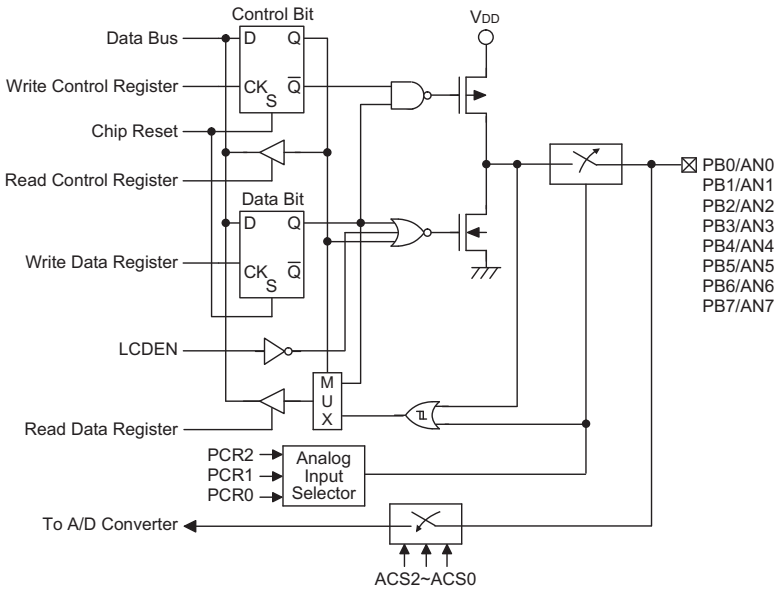
  HT46R94 具有 8 个 A/D 转换输入。所有的模拟输入与 PB 口的 I/O 引脚共用。如果这些引脚被用来作为 A/D 输入而不是一般的 I/O 引脚，则 A/D 转换控制寄存器 ADCR 中相应的位必须被正确的设定。掩膜选项内不包含 A/D 功能。如果这些引脚作为 I/O 引脚使用，仍可以通过掩膜选项选择是否要接上拉电阻。然而如果作为 A/D 输入使用，则这些引脚上的上拉电阻会自动失效。

输入/输出端口结构

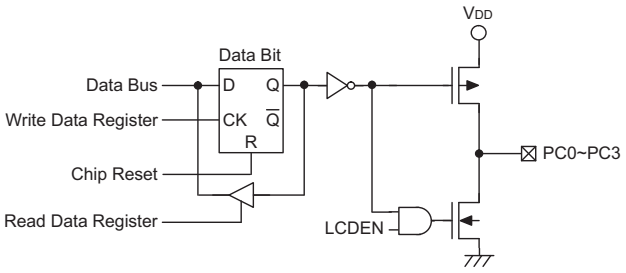
下列为输入/输出端口的内部结构图，可能与芯片内部实际的结构并不完全相同，只是用于帮助用户理解输入/输出端口。



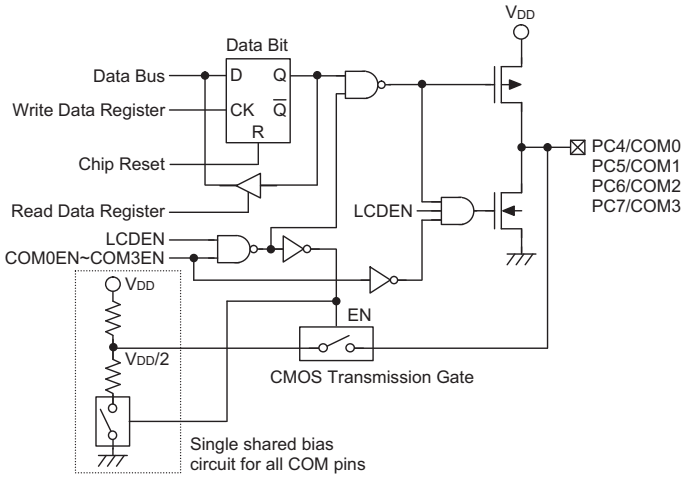
PA 输入/输出端口



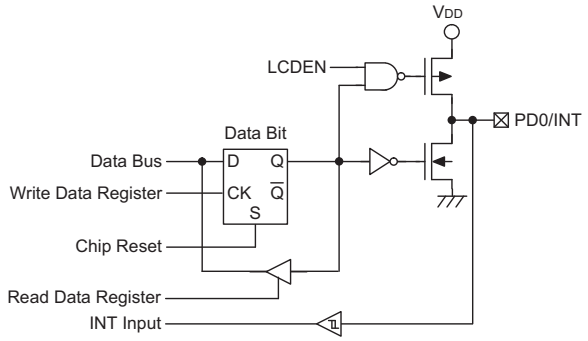
PB 输入/输出端口



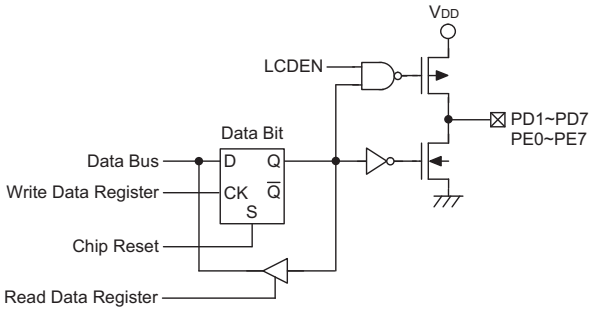
PC0~PC3 输出端口



PC4~PC7 输出端口



PD0 输入/输出端口



PD1~PD7,PE 输出端口

编程注意事项

在使用者的程序中，最先要考虑的是端口的初始化。复位之后，除 PC 端口外，所有的输入/输出数据及端口控制寄存器都将被设为逻辑高。这意味着 PC，PD 和 PE 在输出浮空状态，PA 和 PB 默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉选项。如果 PAC 和 PBC 端口控制寄存器某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口 PA 和 PB 在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或者使用指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意的是当使用这些位控制指令时，一个读-修改-写的操作将会发生。单片机必须先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。

PA 口有唤醒的额外功能，当芯片在 HALT 状态时有很多方法去唤醒此单片机，其中之一就是 PA 口任一个引脚电平由高到低的转换，可以设定 PA 口的一个或多个引脚有这项功能。

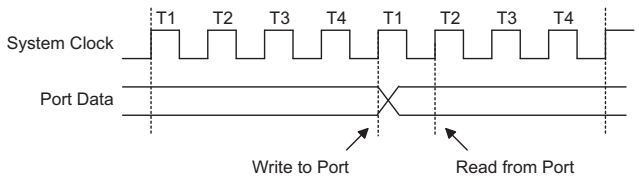
LCD 驱动功能

HT46R94 芯片内部包含外部 LCD 控制电路，功能由 LCDC 控制寄存器实现。

LCD 驱动操作

当 LCDC 寄存器中 LCDEN 位设置为 1，PB 端口设置为输出时，PB、PC，PD 和 PE 为 CMOS 输出，较低的灌/源电流更加适合 LCD 段驱动。上电复位后，PB 端口设置为输入，而 PC 设置为 PMOS 输出，PD 和 PE 设置为 NMOS 输出，适合 LED 驱动。

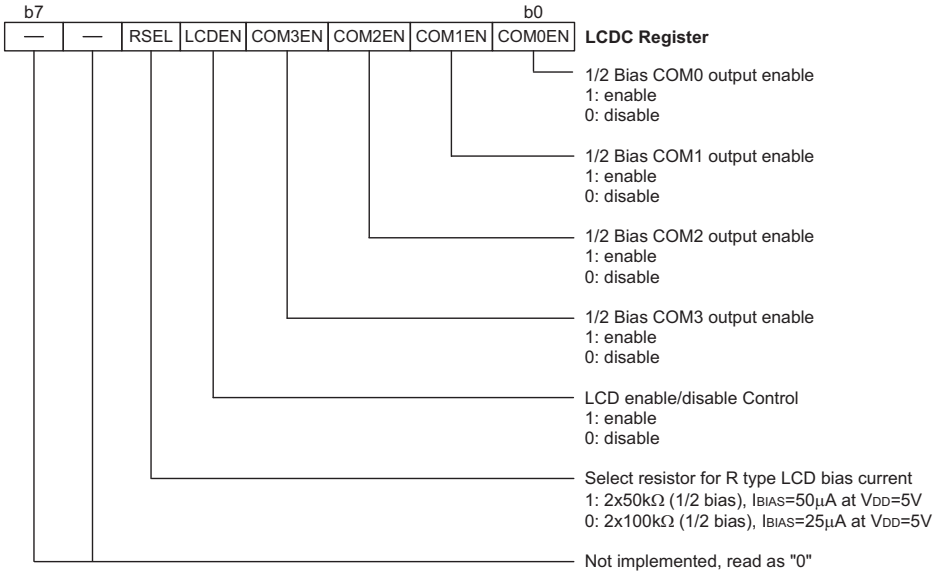
当 LCDEN 位为低，端口寄存器 PC，PD 和 PE 输出为开漏高组态状态；为了设置 PB~PE 为 CMOS 输出，LCDEN 位必须设置为高。特别注意的是，PB 和 PC 端口具有较低灌电流能力（I_{OL2}），而 PD 和 PE 具有较低源电流能力（I_{OH2}），详见直流电气特性表格。



读/写时序

LCD 1/2 偏压控制

由于 HT46R94 可以方便的应用在 LCD 面板驱动场合，引脚 PC4~PC7 和其他输入/输出端口配合使用实现 1/2 偏压 LCD 的时序信号。COM0~3 为引脚 PC4~PC7，而段地址为其他输入/输出端口。内部 1/2 偏压电路的使能通过 LCDC 寄存器中 LCDEN 位和 COM0EN~COM3EN 位实现。LCDC 寄存器中 RSEL 用来设置偏压的电阻值，配合实际 LCD 面板上电阻来最小化电流消耗。注意的是所有的 COM 输出仅共享一个偏压电路。



LCDC 控制寄存器

下面步骤实现 LCD 控制时序：

- 通过设置 RSEL=0 或 1 选择偏压电阻。
- 设置 LCDEN=1。
- 通过软件分别改变 COM 引脚 PC4～PC7 输出高，输出低和输入产生 VDD，VSS，VDD/2 电压。
- 其他输入/输出端口输出低或高电平产生段地址驱动信号。

LCDC 寄存器					1/2 偏压开关
LCDEN	**COM3EN**	**COM2EN**	**COM1EN**	**COM0EN**	
0	x	x	x	x	关闭
1	0	0	0	0	关闭
1	0	0	0	1	打开
: :	: :	: :	: :	: :	打开
1	1	1	1	1	打开

1/2 偏压电路控制

定时/计数器

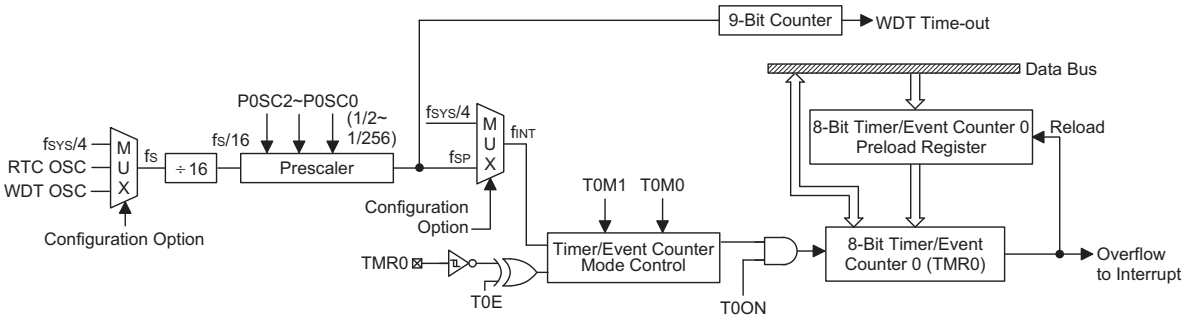
定时/计数器在任何单片机中都是一个很重要的部分，提供程序设计者一种实现和时间有关功能的方法。HT46R94 提供 2 个 8 位的向上定时/计数器。定时/计数器有三种不同的工作模式，可以被当作普通定时器、外部事件计数器、或者脉冲宽度测量器使用。定时器里的 8 级预分频器扩大了定时的范围。

有两个和定时/计数器相关的寄存器，第一个寄存器用来存储实际的计数值，赋值给此寄存器可以设定初始计数值，读取此寄存器可获得定时/计数器的内容。第二个寄存器是定时/计数器的控制寄存器，此寄存器设置定时/计数器的选项，控制定时/计数器的工作模式。定时/计数器的时钟源可掩膜设置来自内部系统时钟或外部引脚输入。

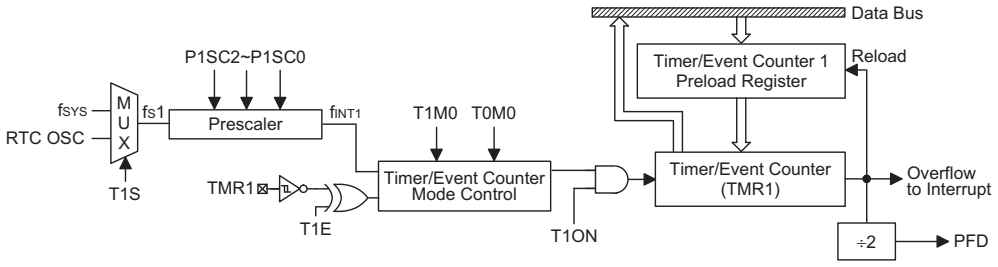
配置定时/计数器输入时钟源

内部定时/计数器时钟源有多种设置，取决于选择哪一个定时/计数器。定时/计数器 0 时钟源通过掩膜选项设定，而定时/计数器 1 通过 TMR1C 控制寄存器中 T1S 位进行设定。当定时/计数器在定时器模式或者在脉冲宽度测量模式时，使用内部时钟作为计时来源。内部时钟还可以通过预分频器进行分频，预分频系数由定时控制寄存器中 PSC0~PSC2 位决定。

定时/计数器在事件计数器模式时使用外部时钟，由外部定时/计数器引脚 TMR0 或 TMR1 提供。每次外部引脚由高电平到低电平或者由低电平到高电平（由 T0E 或 T1E 位决定）跳变时，计数器值加一。



8 位定时/计数器 0 结构



8 位定时/计数器 1 结构

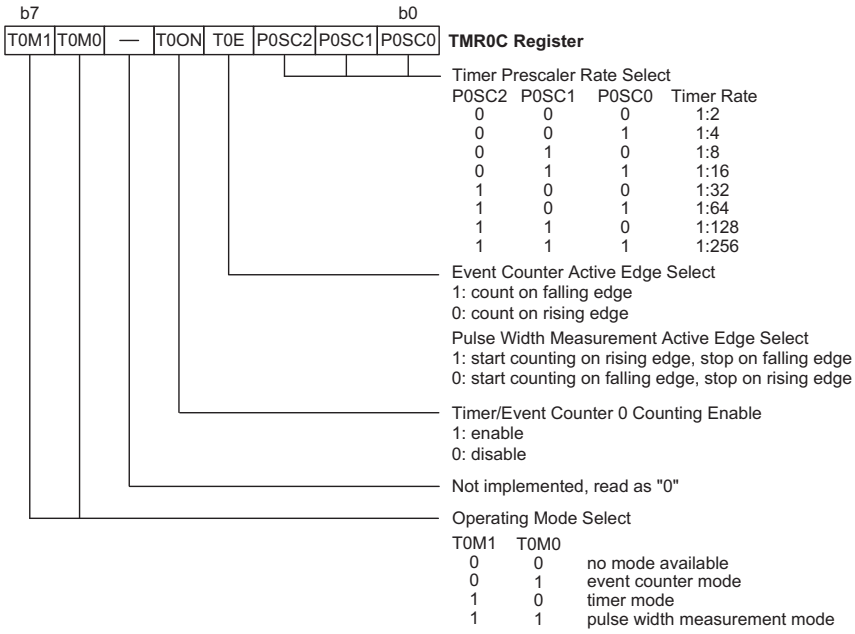
定时/计数寄存器 — TMR0，TMR1

TMR0,TMR1 寄存器位于特殊功能寄存器，用于储存实际定时器值。在用作内部定时且收到一内部计时脉冲时，或用作外部计数且外部定时/计数器引脚状态发生跳变时，此寄存器的值将会加一。定时器将从预置寄存器所载入的值开始计数，直到计满 FFH，此时定时器溢出且会产生一个内部中断信号。定时器自动重新加载计数器初值并继续计数。为了得到定时/计数器 FFH 的最大计数范围，预置寄存器必须先清除为零。注意的是，上电后预置寄存器处于未知状态。定时/计数器在 OFF 条件下，如果把数据写入预置寄存器，这数据将被立即写入实际的定时器。而如果定时/计数器已经 ON 且正在计数，在这个周期内写入到预置寄存器的任何新数据将保留在预置寄存器，只有在下一个溢出发生时才被写入实际定时器。

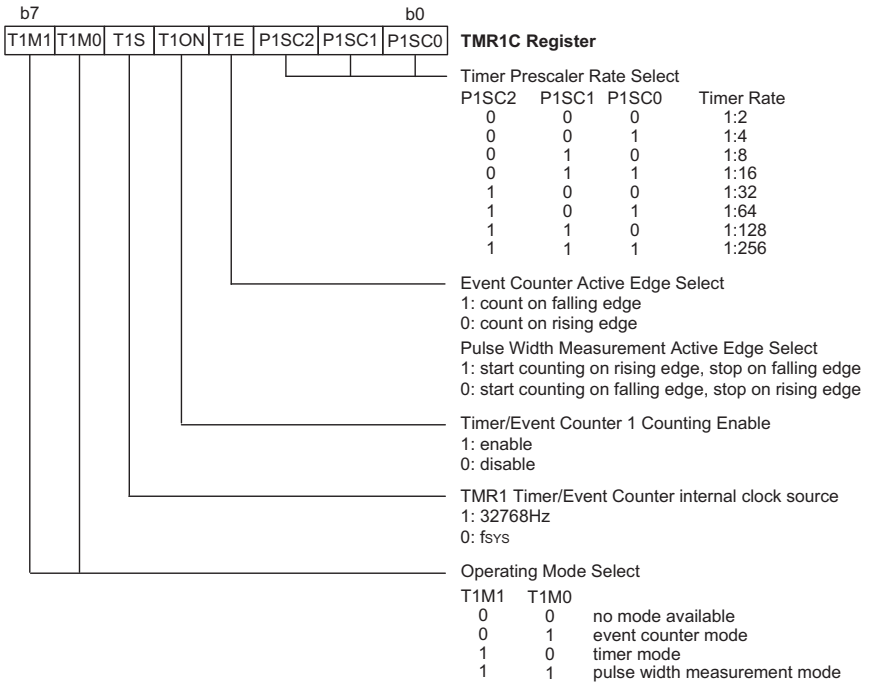
定时/计数控制寄存器 — TMR0C，TMR1C

定时/计数器能工作在三种不同的模式，至于选择工作在哪一种模式则是由各自控制寄存器的内容决定。HT46R94 包含 2 个定时/计数控制寄存器，即 TMR0C 和 TMR1C。定时/计数控制寄存器和对应数据寄存器控制定时/计数器的全部操作。在使用定时器之前，必须正确地设定定时/计数控制寄存器，以便保证定时器能正确操作，而这个过程通常在程序初始化期间完成。

为了确定定时器工作在哪一种模式，定时器模式，外部事件计数模式或脉冲宽度测量模式，定时/计数控制寄存器位 7 和位 6，即 T0M1/T0M0 或 T1M1/T1M0 位必须设定到要求的逻辑电平。定时器打开位 T0ON 或 T1ON，即定时/计数控制寄存器的第 4 位，是定时器控制的开关，设定为逻辑高时，计数器开始计数，而清零时则停止计数。定时/计数控制寄存器的第 0~2 位决定输入定时预分频器中的分频系数。如果使用外部计时源，预分频器的位将不起作用。如果定时器工作在事件计数或脉冲宽度测量模式，T0E 或 T1E 的逻辑电平，即定时/计数控制寄存器的第 3 位将可用来选择上升沿或下降沿触发。



定时/计数控制寄存器 0



定时/计数控制寄存器 1

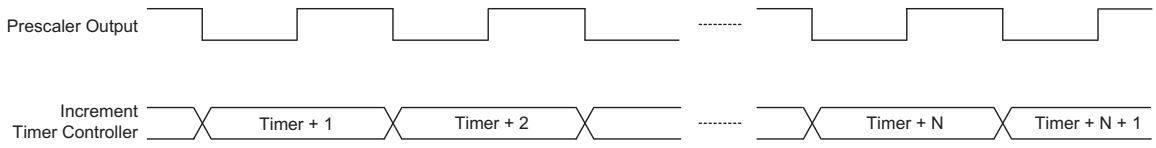
配置定时器模式

在这个模式下，定时器可以用来测量固定时间间隔，当定时器发生溢出时，就会产生一个内部中断信号。要工作在这个模式，TMRC 寄存器中相应位必须设置如下：

位 7	位 6
1	0

定时器模式设置

在这个模式下，定时器的计数源来自内部时钟，取决于所选择的定时/计数器。定时器的输入计时频率是 f_{SYS} 除以定时器预分频器的值，这个值由定时/计数控制寄存器的 0~2 位决定。定时器使能位，即定时/计数控制寄存器位 4 必须被设为逻辑高，才能使定时器工作。每次内部时钟由高到低的电平跳变都会使定时器值增加一。当定时器已满即溢出时，会产生中断信号且定时器会重新载入已经载入到预置寄存器的值，然后继续向上计数。若定时器中断允许，将会产生一个中断信号。寄存器 INTC 中 ETI 位清零，则定时器中断禁止。



定时器模式时序图

配置事件计数模式

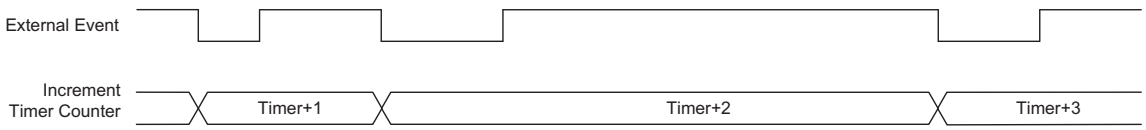
在这个模式下，发生在外部事件计数引脚改变的次数，可以通过内部计数器来记录。为使定时/计数器工作在事件计数器模式，TMRC 寄存器中相应位必须设置如下：

位 7	位 6
0	1

外部事件计数模式设置

系统工作在计数器模式下，时钟源来自外部事件计数引脚，但预分频器不对外部时钟源分频。当设置完定时/计数控制寄存器其余位信息，将计数器使能位，即定时/计数控制寄存器位 4 设为逻辑高，计数器开始计数。当边沿触发位设置为逻辑低，每次外部定时/计数器引脚接收到由低到高的电平跳变将使计数器值加一；而当边沿触发位为逻辑高时，每次外部定时/计数器引脚接收到由高到低的电平跳变将使计数器值加一。若定时中断允许，将会产生一个中断信号。定时器中断可以通过清除 INTC 寄存器中 ETI 位禁止。

由于外部事件计数引脚与输入/输出共用引脚，为了确保定时/计数器工作在事件计数器模式，必须注意两点，首先是将定时/计数控制寄存器中工作模式选择位设定在事件计数器模式，其次是确定端口控制寄存器将这个引脚设定为输入状态。注意的是，在事件计数模式下，即使系统进入 HALT 状态，定时/计数器仍可记录外部引脚的变化。定时/计数器溢出将会唤醒系统，若中断允许将会产生一个定时/计数器中断。



计数器模式时序图

配置脉冲宽度测量模式

在这个模式下，可以测量外部事件引脚上的外部脉冲宽度。系统工作在脉冲宽度测量模式，TMRC 寄存器中相应位必须设置如下：

位 7	位 6
1	1

脉冲宽度测量模式设置

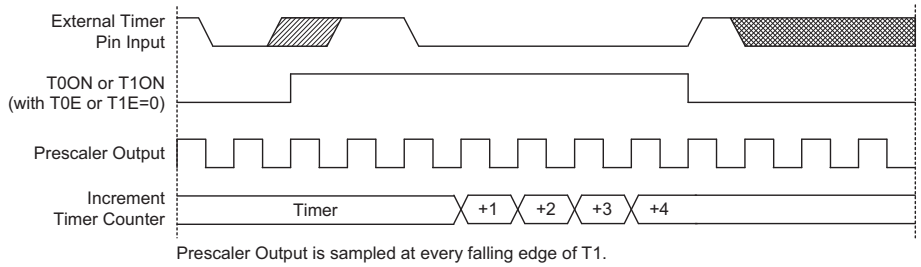
此时，时钟源来自内部时钟，取决于所选择的定时/计数器。系统时钟经内部预分频器分频，分频系数由定时/计数控制寄存器位 0~2 设置。将计数器使能位，即定时/计数控制寄存器位 4 设置为逻辑高，使能内部计数，当外部事件引脚受到有效触发沿时开始计数。

如果有效边沿触发位，即定时/计数控制寄存器位 3 设置为逻辑低，当外部事件引脚接收到一个由高到低的电平跳变时，定时/计数器将开始计数直到外部事件引脚回到它原来的高电平，此时计数器使能位将自动清除为零，且停止计数。而如果有效边沿触发位设置为逻辑高，则当外部事件引脚接收到一个由低到高的电平跳变时，计数器开始计数直到外部事件引脚回到原来的低电平，如上所述，计数器使能位自动清 0，且停止计数。注意的是，在脉冲宽度测量模式中，当外部事件引脚上的外部控制信号回到它原来的电平时，计数器使能位将自动清 0。而在其它两种模式下，定时/计数器使能位只能在程序控制下被清 0。

这时定时/计数器中的值可被程序读取，并由此得知外部定时/计数器引脚接收到的脉冲的长度。当计数器使能位复位时，任何在外部事件计数引脚的电平跳变将被忽略，直到使能位再次被程序设定为逻辑高，计数器才开始脉冲宽度测量。换句话说，用这种方法可完成单个脉冲的测量。

注意的是，在这种模式下，计数器是通过外部事件计数引脚上的逻辑跳变来控制，而不是逻辑电平。当定时/计数器计满产生溢出，定时/计数器将清零并载入预置寄存器的值。若定时器中断允许，将会产生一个内部中断信号。定时/计数器溢出中断可通过中断控制寄存器对应位清 0 来屏蔽。

由于外部事件计数引脚与输入/输出共用引脚，为了确保系统工作在脉冲宽度测量模式，必须注意两点，首先是将定时/计数控制寄存器中工作模式选择位设定在脉冲宽度测量模式，其次是确定此引脚的输入/输出端口控制寄存器对应位被设定为输入状态。



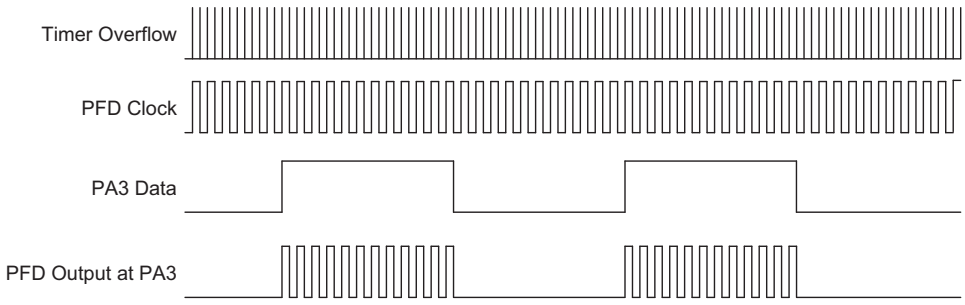
脉冲宽度测量模式时序图

可编程分频器 – PFD

PFD 输出引脚与 PA7 引脚共用。此功能通过掩膜选项选择，如果不选择该功能，则这个引脚作为普通的输入/输出引脚使用。PFD 电路使用定时器 1 溢出信号作为它的时钟源。载入合适的值到定时器预分频器，可以产生需要时钟源的分频系数，由此来控制输出的频率。系统时钟被预分频器分频后的时钟源，进入定时器计时，定时器从预置寄存器的值开始往上计数，直到计数值满而产生溢出信号，导致 PFD 输出改变状态。定时器将自动地重新载入预置寄存器的值，并继续向上计数。

使 PFD 正确运作，必须将 PA 控制寄存器 PAC 位 7 设置为输出。如果把它设置为输入，则 PFD 输出不工作，该引脚仍是作为普通的输入引脚使用。只有将 PA7 置 1，PFD 输出引脚才会有输出。这个输出数据位被用作 PFD 输出的开/关控制。注意，如果 PA7 输出数据位被清 0，PFD 输出将为低电平。

假如系统时钟使用晶体振荡器，则使用这种频率产生的方法可以产生非常精确的频率值。



PFD 输出控制

预分频器

TMR0C 和 TMR1C 寄存器中 P0SC0~P0SC2 和 P1SC0~P1SC2 可以用来定义定时/计数器中内部时钟源的预分频系数。系统工作在外部事件计数模式，预分频器定并无影响。

输入/输出接口

当工作在事件计数或脉冲宽度测量模式时，需要使用外部 PA2/TMR0 或 PA3/TMR1 引脚以确保正确的动作。由于此端口为共用引脚，必须将其设置为定时/计数器的外部输入而不是普通的输入/输出，这需要设置定时/计数控制寄存器工作模式位为外部事件计数或脉宽测量模式。另外 PAC 控制寄存器的第 2 或 3 位必须为高，以保证被设为输入引脚。即使定时/计数器使用了此引脚，掩膜选项的上拉仍有效。

编程注意事项

当定时/计数器工作在定时器模式时，定时器的时钟源使用内部系统时钟且与单片机所有运算都能同步。在这个模式下，当定时器寄存器溢出时，单片机将产生一个内部中断信号，使程序进入相应的内部中断向量。对于脉冲宽度测量模式，定时器的时钟源也是使用内部系统时钟，但定时器只有在正确的逻辑条件出现在外部事件计数引脚时开始工作。由于外部事件和内部定时器时钟无法同步，只有当下一个定时器时钟到达时，单片机才会发现这个外部事件，因此在测量值上可能有小的差异，需要程序设计者在程序应用时加以注意。同样的情况发生在定时/计数器配置为外部事件计数模式时，它的时钟源是外部事件，与内部系统时钟或者定时器时钟不同步。

当读取定时/计数器或写数据到预置寄存器时，计数时钟会被阻隔以避免错误发生，但这样做可能会导致计数错误，所以程序设计者应该考虑到这点。在第一次使用定时/计数器之前，要仔细确认有没有正确地设定初始值。中断控制寄存器中的定时器使能位必须正确的设置，否则相应定时/计数器内部中断仍然无效。定时/计数器控制寄存器中的边沿触发选择、定时/计数器工作模式和时钟源控制位也必须正确的设定，以确保定时/计数器按照应用需求而正确的配置。在定时/计数器打开之前，必须确保先载入定时/计数器寄存器的初始值，这是因为在上电后，定时/计数器寄存器中的初始值是未知的。定时/计数器初始化后，可以使用定时/计数器控制寄存器中的使能位来打开或关闭定时器。注意的是，在打开定时器之前，必须先确定定时器模式是否已设定。定时/计数器使能位和工作模式的修改同时设置，可能会导致错误结果。

当定时/计数器发生溢出，相应的中断请求标志将置位。若中断允许，将会产生一个中断信号。不管中断是否允许，在 HALT 状态下，定时/计数器的溢出也会产生唤醒，这种情况可能发生在外部信号变化的计数模式中，定时/计数器向上计数直至溢出并唤醒系统。若在 HALT 模式下，不需要定时器中断唤醒系统，可以在进入 HALT 前将相应中断请求标志位置 1。

定时/计数器应用范例

这个例子说明了如何设置定时/计数器 0 的寄存器，如何设置和控制中断。另外还需注意的是，怎样通过 TMR0C 寄存器的第 4 位来启停定时/计数器。此应用范例设置定时/计数器 0 为定时模式，时钟来源于内部的系统时钟。

```
org 04h                    ; external interrupt vectors
reti
org 08h                    ; timer/event counter 0 interrupt vector
jmp tmrint0                ;jmp here when timer/event counter 0 overflows
    :
org 20h                    ; main program
                           ; internal timer/event counter 0 interrupt routine
tmrint0:
                           ;timer/event counter 0 main program placed here
    :
reti
    :
    :
begin:
                           ; setup timer registers
mov a,09bh                 ; setup timer preload value
mov tmr0,a
mov a,081h                 ; setup timer control register
mov tmr0c,a                ; timer mode and prescaler set to /4
                           ; setup interrupt register
mov a,005h                 ; enable master interrupt and timer interrupts
mov intc0,a
set tmr0c.4                ;start timer/event counter0– note mode bits must be previously setup
```

脉冲宽度调制器

HT46R94 提供 3 个脉冲宽度调制输出，即 PWM0，PWM1 和 PWM2。这在马达速度控制等应用中十分有用，通过给相应的 PWM 寄存器设置特殊的值，PWM 功能可提供占空比可调而频率固定的波形。

通道	PWM 模式	输出引脚	寄存器名称
3	6+2 或 7+1	PA4 PA5 PA6	PWM0 PWM1 PWM2

PWM 简介

PWM 输出通过掩膜选项进行设置。PWM0，PWM1 和 PWM2 寄存器位于数据存储器，由 8 位组成，表示输出波形中每个调制周期的占空比。为了提高 PWM 调制频率，每一个调制周期被调制 2 个或 4 个独立的调制子区段，即 7+1 模式或 6+2 模式，所需工作模式通过掩膜选项进行设置。注意的是，当使用 PWM 时，只要将所需的值写入 PWM0，PWM1 或 PWM2 寄存器内，单片机的内部硬件会自动地将波形细分为子调制周期。PWM 时钟源就是系统时钟 f_{SYS}。

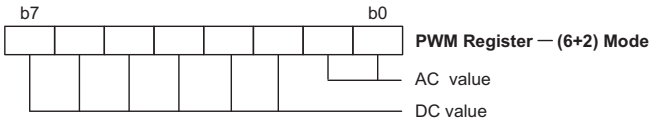
将原始调制周期分成 2 个或 4 个子周期的方法，使产生更高的 PWM 频率成为可能，这样可以提供更广泛的应用。只要产生的 PWM 脉冲周期小于负载的时间常数，PWM 输出就比较合适，这是因为长时间常数负载将会平均 PWM 输出的脉冲。使用者必须理解 PWM 频率与 PWM 调制频率的不同之处。当 PWM 时钟为系统时钟 f_{SYS}，而 PWM 值为 8 位时，整个 PWM 周期的频率为 $f_{SYS}/256$。然而工作在 7+1 模式时，PWM 调制频率将会是 $f_{SYS}/128$；工作在 6+1 模式时，PWM 调制频率将会是 $f_{SYS}/64$。

6+2 PWM 模式

通过一个 8 位的 PWM 寄存器控制，每个完整的 PWM 周期由 256 个时钟周期组成。在 6+2 PWM 模式中，每个 PWM 周期又被分成 4 个独立的子周期，称为调制周期 0~调制周期 3，在表格中以“i”表示。4 个子周期各包含 64 个时钟周期。在这个模式下，得到以 4 为因数增加的调制频率。8 位的 PWM 寄存器被分成两个部分，这个寄存器的值表明整个 PWM 波形的占空比。第一部分包括第 2 位~第 7 位，表示 DC 值，第二部分为第 0 位~第 1 位，表示 AC 值。在 6+2 PWM 模式中，4 个调制子周期的占空比，分别如下表所示。

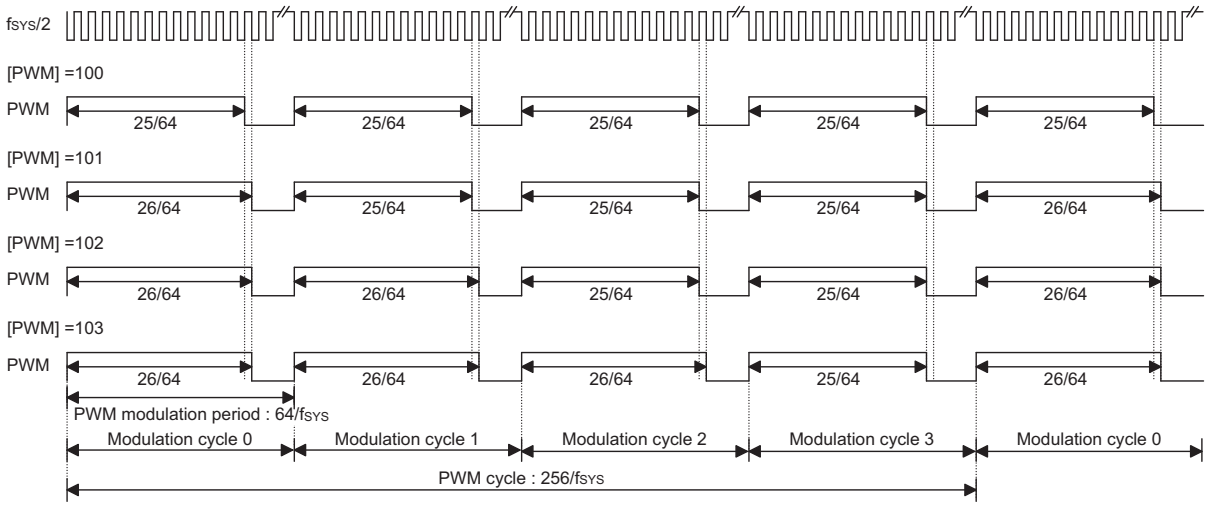
参数	AC (0~3)	DC (占空比)
调制周期 i (i=0~3)	i<AC	$\frac{DC+1}{64}$
	i≥AC	$\frac{DC}{64}$

6+2 模式调制周期值



6+2 模式 PWM 寄存器

下图表示 6+2 模式下 PWM 输出的波形。请特别注意单个的 PWM 周期是如何分成 4 个独立的调制周期以及 AC 值与 PWM 值的关系。



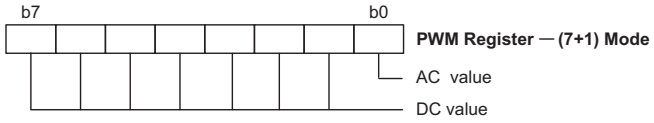
6+2 PWM 模式

7+1 PWM 模式

通过一个 8 位的 PWM 寄存器控制，每个完整的 PWM 周期由 256 个时钟周期组成。在 7+1 PWM 模式中，每个 PWM 周期又被分成 2 个独立的子周期，称为调制周期 0~调制周期 1，在表格中以“i”表示。2 个子周期各包含 128 个时钟周期。在这个模式下，得到以 2 为因数增加的调制频率。8 位的 PWM 寄存器被分成两个部分，这个寄存器的值表明整个 PWM 波形的占空比。第一部分包括第 1 位~第 7 位，表示 DC 值，第二部分为第 0 位，表示 AC 值。在 7+1 PWM 模式中，2 个调制子周期的占空比，分别如下表所示。

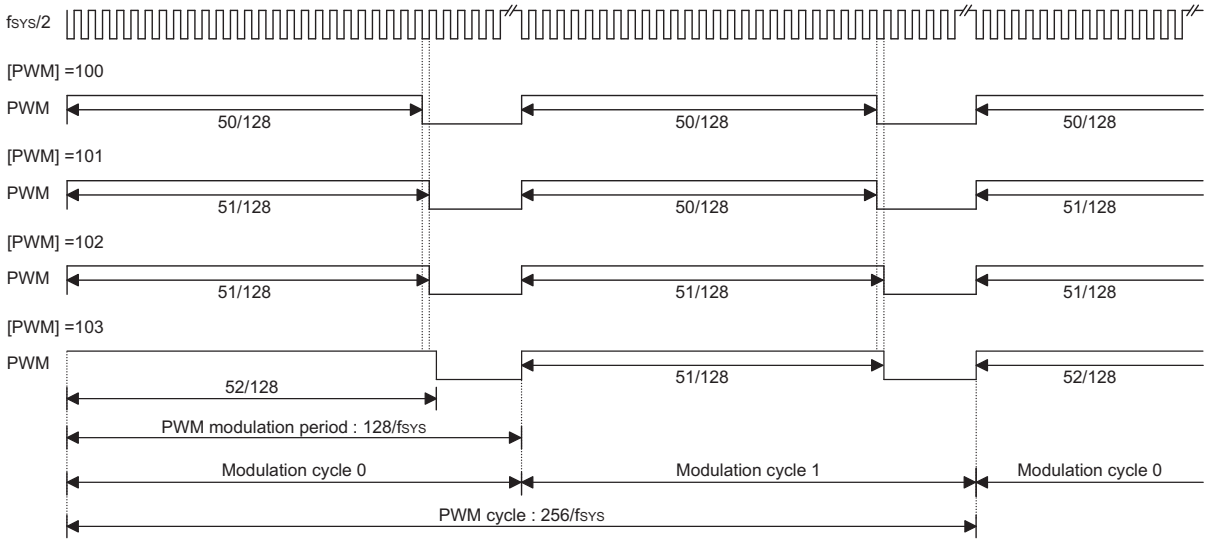
参数	AC (0~1)	DC (占空比)
调制周期 i (i=0~1)	i<AC	$\frac{DC+1}{128}$
	i≥AC	$\frac{DC}{128}$

7+1 模式调制周期值



7+1 模式 PWM 寄存器

下图表示 7+1 模式下 PWM 输出的波形。请特别注意单个的 PWM 周期是如何分成 2 个独立的调制周期以及 AC 值与 PWM 值的关系。



7+1 PWM 模式

PWM 输出控制

PWM0 输出与 PA4 引脚共用，PWM1 输出与 PA5 引脚共用，PWM2 输出与 PA6 引脚共用。使引脚作为 PWM 输出而非普通的 I/O 引脚，掩膜选项必须正确设置。I/O 端口控制寄存器 PAC.4，PAC.5 或 PAC.6 也必须写 0，以确保所需要的 PWM0，PWM1 或 PWM2 输出引脚设置为输出状态。在完成这两个初始化步骤，以及将所要求的 PWM 值写入 PWM0，PWM1 或 PWM2 寄存器之后，将 PA.4，PA.5 或 PA.6 数据寄存器写入 1，则 PWM 数据就会出现在引脚上；将 PA.4，PA.5 或 PA.6 数据寄存器写入 0，则会除能 PWM 输出功能并强制输出低电平。也就是说，PA.4，PA.5 或 PA.6 作为 PWM 功能的开/关控制来使用。需要注意的是，即使掩膜选项设置 PWM 功能，但 PAC 控制寄存器中相应的位又写入 1，使其成为输入引脚，则此引脚仍是作为带上拉电阻的正常输入端使用。

PWM 调制频率	PWM 频率	PWM 占空比
$f_{SYS}/64$	$f_{SYS}/256$	[PWM]/256

PWM 应用范例

下面的范例程序表明如何设置和控制 PWM 输出，应用时必须在掩膜选项中设置 PWM 功能。

```
mov     a,64h           ;setup PWM0 value of 100,decimal which is 64H
mov     pwm0,a
clr     pac.4           ;setup pin PA4 as output
set     pa.4            ; pa.4=1;enable the PWM output
    :
clr     pa.4            ; disable the PWM output – PA4 will remain low
```

A/D 转换器

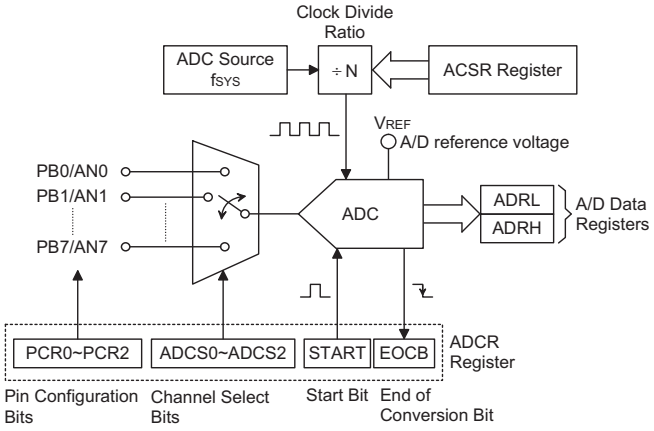
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先必须通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有低成本和减少器件空间需求的优势。

A/D 简介

HT46R94 包含了 8 通道的 A/D 转换器，它们可以直接接入外部模拟信号（来自传感器或其它控制信号）并直接将这些信号转换成 12 位的数字量。

输入通道	转换位数	输入引脚
8	12	PB0~PB7

下图显示了 A/D 转换器内部结构和相关的寄存器。



A/D 转换器结构

A/D 转换数据寄存器 — ADRL，ADRH

对于具有 12 位的 A/D 转换器的芯片，需要一个高字节寄存器 ADRH 和一个低字节寄存器 ADRL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。只有高位寄存器 ADRH 完全利用了 8 位，低位寄存器 ADRL 只利用了 8 位中的 4 位包含转换值。在下表中，D0~D11 是 A/D 换转数据结果位。

寄存器	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADRL	D3	D2	D1	D0	—	—	—	—
ADRH	D11	D10	D9	D8	D7	D6	D5	D4

A/D 数据寄存器

A/D 转换器控制寄存器 – ADCR

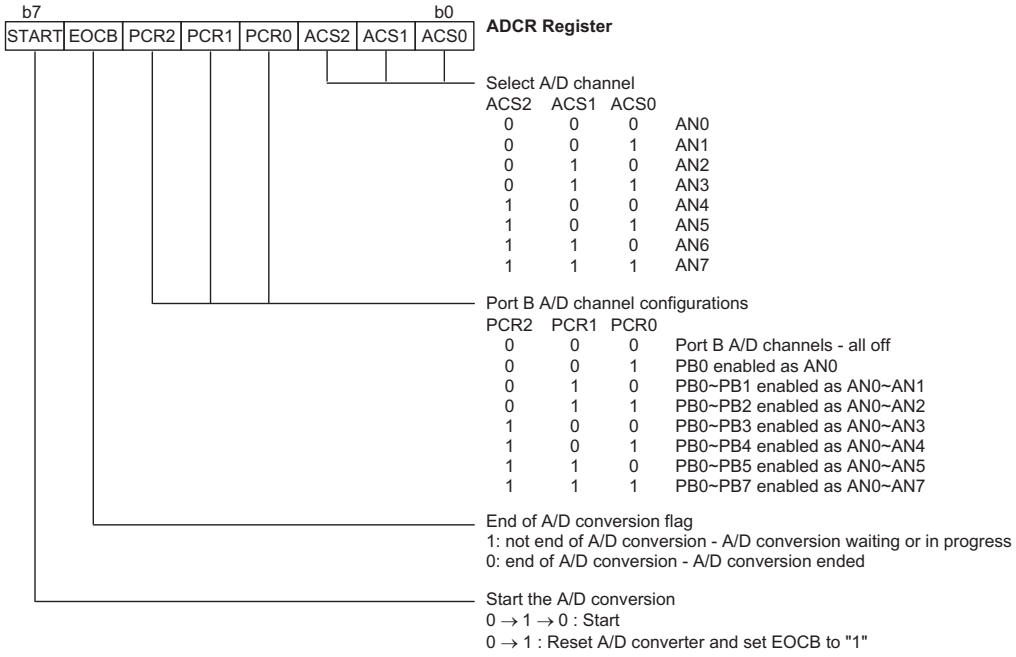
寄存器 ADCR 用来控制 A/D 转换器的功能和操作。这个 8 位的寄存器所定义的功能包括选择哪一个模拟通道连接至内部 A/D 转换器，哪个引脚是模拟输入，哪个引脚是正常 I/O，并控制和监视 A/D 转换器的开始和复位功能。

寄存器 ADCR 包含 ACS2~ACS0 位，它们定义通道的编号。由于每个单片机只包含一个实际的模数转换电路，因此 8 通道模拟输入中每一个都必须分别被连接转换器，ADCR 寄存器中 ACS2~ACS0 位的功能决定哪个模拟通道真正连接到内部 A/D 转换器。

ADCR 寄存器中的 PCR2~PCR0 位，用来定义 PB 端口上哪些引脚为 A/D 转换器的模拟输入，哪些引脚为正常的 I/O。如果 PCR2~PCR0 这 3 位的值为 111，也就是 AN0~AN7 都将被设定为模拟输入。注意的是，如果 PCR2~PCR0 全都设为 0，则 PB 端口的引脚都被设定为正常的 I/O，这时内部 A/D 转换电路的电源将被关闭以减少功耗。

ADCR 寄存器中的 START 位，用于打开和复位 A/D 转换器。当单片机设定此位从逻辑低到逻辑高，然后再到逻辑低，就会开始一个模数转换周期。当 START 位从逻辑低到逻辑高，但不再回到逻辑低时，ADCR 寄存器中的 EOCB 位置 1，复位模数转换器。START 位用于控制内部模数换转器的开/关动作。

ADCR 寄存器中的 EOCB 位用于表明模数转换过程的完成。在转换周期结束后，EOCB 位会被单片机自动地置为 0。此外，也会置位中断控制寄存器内相应的 A/D 中断请求标志位，如果中断使能，就会产生适当的内部中断信号，A/D 内部中断信号将引导程序到相应的 A/D 内部中断入口；如果 A/D 内部中断被禁止，单片机会轮询 ADCR 寄存器中的 EOCB 位，检查此位是否被清除，以作为另一种侦测 A/D 转换周期结束的方法。



A/D 转换控制寄存器

A/D 转换功耗控制

由于 A/D 转换电路内建在芯片中，A/D 转换过程中将消耗部分电能。但是系统提供两种方法供用户打开/关闭 A/D 转换，以降低功耗。一种方法是将 ADCR 寄存器中 PCR0~PCR2 清 0，以关闭 A/D 转换；另外一种方法是将 ACSR 寄存器中 ADONB 位设置为 1，以关闭 A/D 转换。两种方法都可以独立使用以降低功耗。

值得注意的是，A/D 转换电路电源由 VDD 引脚提供。当 A/D 转换电路中模拟部分最低工作电压高于数字部分工作电压时将导致模拟部分无法在 VDD 规格下工作。芯片直流电气特性表格中单独列出了模拟电路工作电压规格。

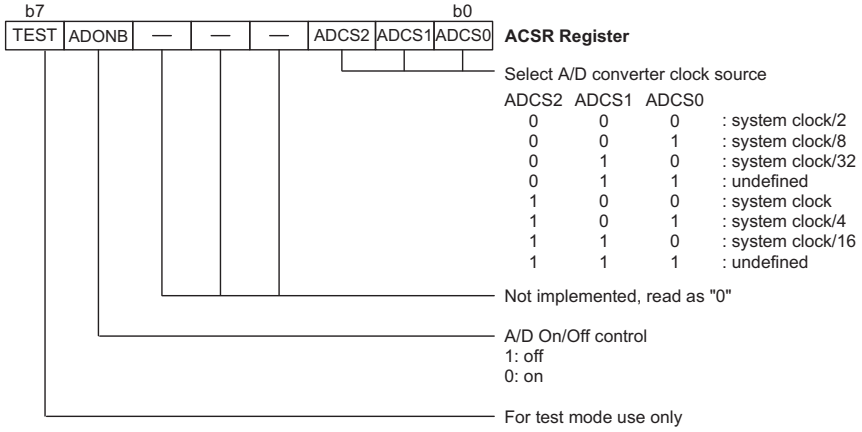
PCR 位	ADONB 位	A/D 转换电路
0	X	关闭
＞0	0	打开
＞0	1	关闭

A/D 功耗控制

A/D 转换时钟源寄存器 –ACSR

A/D 转换时钟源为系统时钟 f_{SYS} 的分频，分频系数由 ACSR 寄存器中 ADCS1 和 ADCS0 位共同决定。

虽然 A/D 转换的时钟源是由系统时钟 f_{SYS}、ADCS1 和 ADCS0 共同决定，但可选择的最大 A/D 时钟源速度则有一些限制。由于单片机允许的 A/D 时钟周期 t_{AD} 的最小值为 0.5μs，因此，当系统时钟速度超过 4MHz 时，ADCS1 和 ADCS0 位设定值不能设置为 00，否则将会产生不准确的 A/D 转换值。参考下面表格中的一些例子，被标上星号*的数值是不允许的，因为它们的 A/D 时钟周期小于规定的最小值。



A/D 转换时钟控制寄存器

f_{SYS}	A/D 时钟周期(t_{AD})			
	ADCS2~ADCS0 =000（f_{SYS}/2）	ADCS2~ADCS0 =001（f_{SYS}/8）	ADCS2~ADCS0 =010（f_{SYS}/32）	ADCS2~ADCS0 =011
1MHz	2μs	8μs	32μs	未定义
2MHz	1μs	4μs	16μs	未定义
4MHz	500ns*	2μs	8μs	未定义
8MHz	250ns*	1μs	4μs	未定义

A/D 时钟周期范例

A/D 输入引脚

所有的 A/D 模拟输入引脚都与 PB 端口的 I/O 引脚共用。ADCR 寄存器中的 PCR2~PCR0 位，决定是将输入引脚设置为普通的 PB 输入/输出引脚，还是设置为模拟输入引脚，而不是通过掩膜选项设定。也就是说，引脚的功能可由程序来控制，从普通 I/O 操作功能到模拟输入，反过来也一样。当输入引脚作为普通 I/O 引脚使用时，可通过掩膜选项设置上拉电阻；若设置为 A/D 输入，则上拉电阻会自动断开。注意的是，PB 端口控制寄存器并不需要为使能 A/D 输入，而先设定 A/D 引脚为输入引脚，当 PCR2~PCR0 位使能 A/D 输入时，不考虑端口控制寄存器的状态。A/D 转换器的参考电压由单独的 VREF 引脚提供，可以和外部基准电压源连接。实际测量时模拟输入电压 VREF 不可超过 VDD 电压值。另外必须确保 VREF 电压的稳定及减少噪声。

A/D 转换初始化

内部的 A/D 转换器必须经过一个特殊的方法初始化。当转换通道选择位改变时，A/D 转换器必须重新初始化。如果通道选择位改变后没有重新初始化，那么 EOCB 标志为可能会处于不确定的值，这样可能会导致一个错误的转换结束信号。当通道选择位改变后，寄存器 ADCR 中的 START 位必须在 1 到 10 个指令周期内先置位再立即清零，这样可以保证 EOCB 被正确的置位。

A/D 转换步骤

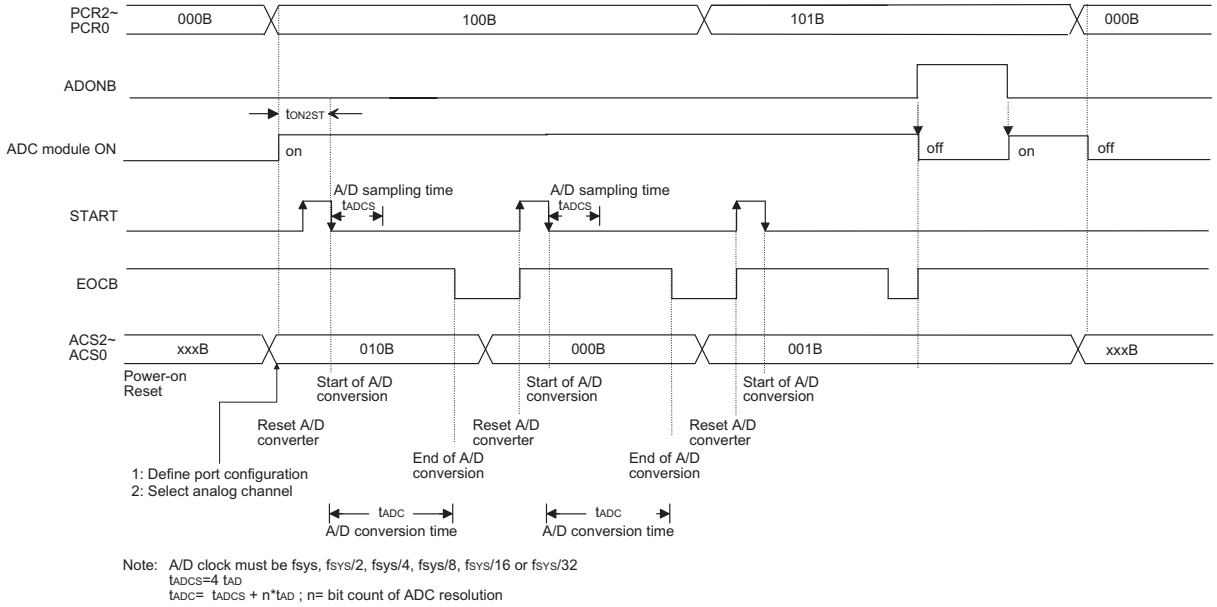
下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
  通过设置 ACSR 寄存器中的 ADCS1 和 ADCS0 位，选择所需的 A/D 转换时钟。
- 步骤 2

  通过将 ACSR 寄存器中的 ADONB 位清 0，使能 A/D 转换。
- 步骤 3
  通过设置 ADCR 寄存器中的 ACS2~ACS0 位，选择连接至内部 A/D 转换器的通道。
- 步骤 4

  选择 PB 端口的 A/D 输入引脚，并通过 ADCR 寄存器中的 PCR2~PCR0 位设置为 A/D 输入引脚。此步骤也可在第二步写 ADCR 寄存器时完成。
- 步骤 5
  如果要使用中断，则中断控制寄存器必须正确地设置，以确保 A/D 功能的动作。中断控制寄存器里总中断控制位 EMI 必须置位和 A/D 转换器的中断使能位 EADI 都必须置位为“1”。
- 步骤 6
  通过设定 ADCR 寄存器中的 START 位从“0”到“1”再回到“0”，可以开始模数转换的过程。该位需初始化为“0”。
- 步骤 7
  轮询 ADCR 寄存器中的 EOCB 位，检查模数转换过程是否完成。当此位成为逻辑低时，表示转换过程已经完成。转换完成后，可读取 A/D 数据寄存器 ADRL 和 ADRH 获得转换后的值。若中断使能且堆栈未满，则转换完成后程序会进入 A/D 中断服务子程序也是一种可选方式。
  注：若使用轮询 ADCR 寄存器中 EOCB 位的状态的方法来检查转换过程是否结束时，步骤 5 可以省略。

下列时序图表示模数转换过程中不同阶段的图形与时序。



A/D 转换时序图

A/D 转换器没有对应的掩膜选项设定，它的功能设定与操作完全由应用程序控制。由应用程序控制开始 A/D 转换过程后，单片机的内部硬件就会开始进行转换，在这个过程中，程序可以继续其它功能。A/D 转换时间为 $16t_{AD}$，t_{AD} 为 A/D 转换所需要的周期。

编程注意事项

在编写程序时，必须特别注意寄存器 ADCR 中的 A/D 转换通道选择位。如果这些全部为零，那么没有外部引脚连接到 A/D 转换器，此时的外部引脚可作为普通的 IO 口使用。这样可以减少 A/D 转换部分的功耗，对电池供电的系统非常重要。A/D 转换器也可以通过 ACSR 寄存器中 ADONB 位关闭。

另一个编程注意事项是，当 A/D 转换通道选择位改变后，A/D 转换器必须重新初始化。当通道选择位改变后，给寄存器 ADCR 的 START 位一个脉冲即可初始化 A/D 转换器。只有选择位全部被清零，A/D 转换不需要重新初始化。

A/D 转换应用范例

下面两个范例程序用来说明怎样使用 A/D 转换。第一个范例是轮询 ADCR 寄存器中的 EOCB 位来判断 A/D 转换是否完成；第二个范例则使用中断的方式判断。

范例：使用 EOCB 轮询方法侦测转换的结束

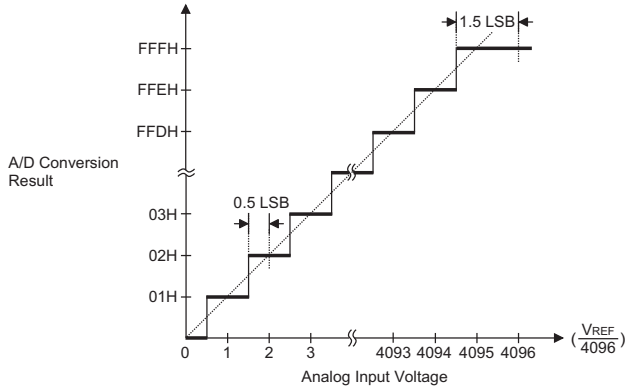
```
        clr EADI                ; disable ADC interrupt
        mov a,00000001B
        mov ACSR,a              ; setup the ACSR register to select fSYS/8 as the A/D clock
        mov a,00100000B         ; setup ADCR register to configure Port PB0~PB3
                                ; as A/D inputs and select input AN0 to be
        mov ADCR,a              ; connected to the A/D converter
            :                   ; As the Port A channel bits have changed the following
                                ; START signal (0-1-0) must be issued within 10 instruction cycles
Start_conversion:
        clr START
        set START               ; reset the A/D
        clr START               ; start the A/D
Polling_EOC:
        sz EOCB                 ; poll the ADCR register EOCB bit to detect end of A/D conversion
        jmp polling_EOC         ; continue polling
        mov a,ADRL               ; read low byte conversion result value
        mov adrl_buffer,a       ; save result to user defined memory
        mov a,ADRH              ; read high byte conversion result value
        mov adrh_buffer,a       ; save result to user defined memory

            :
        jmp Start_conversion    ; start next A/D conversion
```

范例：使用中断方式侦测转换的结束

```
        clr EADI                ; disable ADC interrupt
        mov a,00000001B
        mov ACSR,a              ; setup the ACSR register to select fSYS/8 as the A/D clock
        mov a,00100000B         ; setup ADCR register to configure Port PB0~PB3
                                ; as A/D inputs and select input AN0 to be
        mov ADCR,a              ; connected to the A/D converter
            :                   ; As the Port A channel bits have changed the following
                                ; START signal (0-1-0) must be issued within 10 instruction cycles
            :
Start_conversion:
        clr START
        set START               ; reset the A/D
        clr START               ; start the A/D
        clr ADF                 ; clear ADC interrupt request flag
        set EADI                ; enable ADC interrupt
        set EMI                 ; enable global interrupt
            :
            :
ADC_ISR:                        ; ADC interrupt service routine
        mov acc_stack,a         ; save ACC to user defined memory
        mov a,STATUS
        mov status_stack,a      ; save STATUS to user defined memory
            :
            :
        mov a,ADRL               ; read low byte conversion result value
        mov adrl_buffer,a       ; save result to user defined memory
        mov a,ADRH              ; read high byte conversion result value
        mov adrh_buffer,a       ; save result to user defined memory
            :
            :
EXIT_INT_ISR:
        mov a,status_stack
        mov STATUS,a
        mov a,acc_stack
        reti
```

A/D 转换功能

HT46R94 含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于 VDD 的电压值，因此每一位可表示 V_{DD}/4096 的模拟输入值。下图显示 A/D 转换器模拟输入值和数字输出值之间理想的转换功能。



理想的 A/D 转换功能

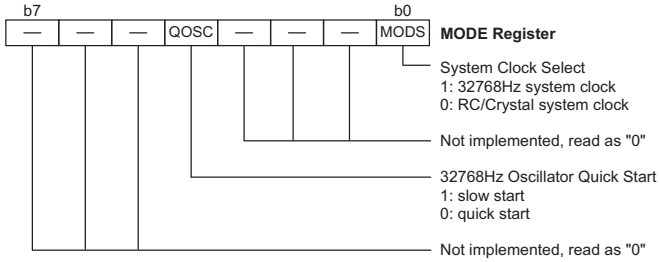
为了减少量化错误，A/D 转换器输入端会加入 0.5LSB 的偏移量。除了数字化数值 0，其后的数字化数值会在精确点之前的 0.5LSB 处改变，而数字化数值的最大值将在 VDD 之前的 1.5 LSB 处改变。

系统工作模式

HT46R94 可以工作在正常模式，慢速模式和暂停模式。为了使单片机工作在不同的模式，系统必须同时提供两种时钟，OSC1 和 OSC2 连接 RC 或晶体振荡器，OSC3 和 OSC4 连接 32768Hz 晶体振荡器。

模式选择

系统时钟选择 RC/XTAL 或 32768Hz RTC 可以通过 MODE 寄存器中 MODS 位进行设置，将 MODS 位设置为 1，即选择 32768Hz 晶体振荡器的慢速模式可降低系统功耗。在慢速模式下，RC 或晶体振荡器将关闭。将 MODE 寄存器中 MODS 位清 0，系统工作在较高频率的正常模式。剩下的的暂停模式是通过 HALT 指令实现。注意的是，在所有的工作模式中 32768Hz 晶体振荡器保持运行。



系统工作模式寄存器-MODE

如果将 32768Hz 晶体振荡器设置为系统时钟，看门狗时钟源的掩膜选项必须设置为 32768Hz 晶体振荡器，否则系统将出现不可预料的错误。

模式	系统时钟	HALT 指令	MODS	RC/XTAL 振荡器	32768Hz
正常	RC/XTAL 振荡器	不执行	0	打开	打开
慢速	32768Hz	不执行	1	关闭	打开
暂停	HALT	执行	X	关闭	打开

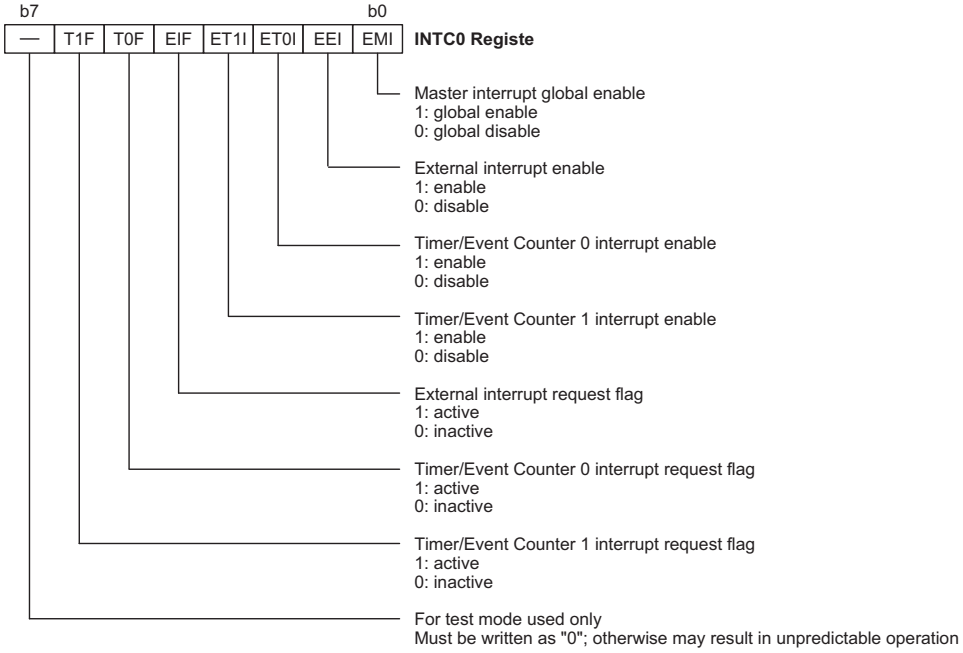
工作模式

中断

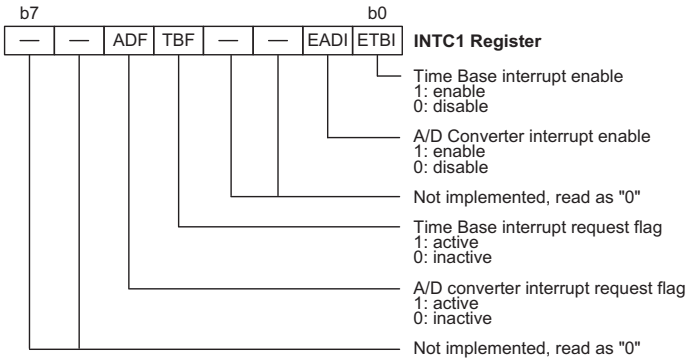
中断是单片机一个重要功能。当外部中断或内部中断如定时/计数器，时基或 A/D 转换有效，系统会中止当前的程序，而转到相对应的中断服务程序。此系列每一款单片机均提供一个外部中断和多种内部中断，外部中断与 INT 引脚相关，而内部中断与定时/计数器和时基溢出和 A/D 转换相关。

中断寄存器

所有中断允许和请求标志均由数据存储器内 INTC0 和 INTC1 寄存器控制。通过控制相应的中断允许位使能或禁止相关的中断。当发生中断，相应有中断请求标志将被置位。总中断请求标志清零将禁止所有中断允许。



中断控制寄存器 0

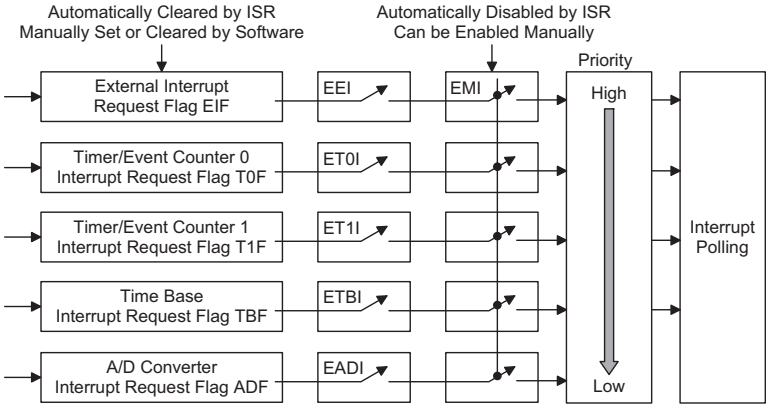


中断控制寄存器 1

中断操作

如果相应的中断允许，定时/计数器溢出、时基溢出，A/D 转换结束或外部中断引脚有效沿触发都会产生一个中断请求，下条指令的地址将被压入堆栈，相应的中断向量地址加载至 PC 中，系统将从此向量取下条指令。中断向量处通常为跳转指令，以跳转到相应的中断服务程序。中断服务程序必须以 RETI 指令返回至主程序，以执行原来的程序。

各个中断使能位以及相应的请求标志位，以优先级的顺序如下图所示。



中断结构图

一旦中断子程序被响应，系统将自动清除 EMI 位，所有其它的中断将被屏蔽，这个方式可以防止任何进一步的中断嵌套。如果其它中断请求可能发生在此期间，只有中断请求标志位会被记录。如果某个中断服务子程序正在执行，此时有另一个中断要求响应，EMI 位可以被置位，以允许此中断被响应。如果堆栈已满，即使此中断使能，中断请求也不会被响应，直到 SP 减少为止。如果要求立刻动作，则堆栈必须避免成为储满状态。

中断优先级

中断发生在两个连续的 T2 脉冲上升沿之间，如果相应的中断请求被允许，中断将在后一个 T2 脉冲响应。下表指出同时提出请求的情况下所提供的优先级，可以通过重新设定 EMI 位来加以屏蔽。

中断源	中断向量	优先级
外部中断	04H	1
定时/计数器 0 溢出中断	08H	2
定时/计数器 1 溢出中断	0CH	3
时基溢出中断	10H	4
A/D 转换中断	14H	5

假使外部和内部中断均被使能，且外部和内部中断同时发生，则外部中断永远优先处理，首先被响应。使用 INTC0 和 INTC1 控制寄存器适当地屏蔽个别中断，可以防止中断同时发生的情况。

外部中断

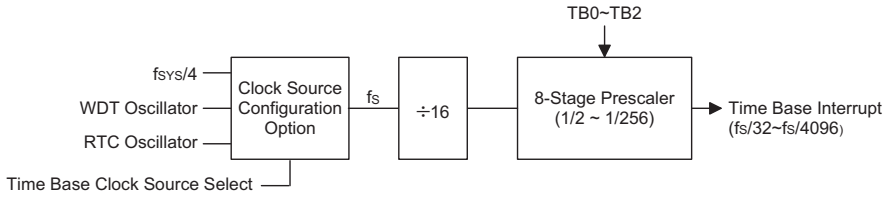
使外部中断发生，总中断控制位 EMI、外部中断使能位 EEI 必须先被置位。当外部中断有效触发，INTC0 寄存器中 EIF 位将被设定。由于外部中断可以双边沿触发，当 INT 引脚出现上升沿触发或下降沿触发时，EIF 标志位将被设置为 1。外部中断和 PD0 共享引脚，使用外部中断时必须将 INTC0 寄存器中外部中断使能位设置为 1，LCDC 寄存器中 LCDEN 位设置为 0，同时将 PD0 位设置为 1；如果外部中断使能位未设置为 1，共享引脚将作为 PD0 的 CMOS 输出。当中断使能、堆栈未满且外部中断产生时，将调用位于地址 04H 处的子程序。当响应外部中断服务子程序时，中断请求标志位 EIF（即 INTC0.4）会被复位且 EMI 位会被清零以除能其它中断。

定时/计数器中断

使定时/计数器中断发生，总中断控制位 EMI、INTC0 寄存器中定时/计数器中断使能位 ET0I 和 ET1I 必须先被置位。当定时/计数器发生溢出，INTC0 寄存器中 T0F 或 T1F 中断请求标志位置位并触发定时/计数器中断。若中断使能，堆栈未满，当发生定时/计数器中断时，将调用位于地址 08H 或 0CH 处的子程序。当响应定中断服务子程序时，中断请求标志位 T0F 或 T1F 会被复位且 EMI 位会被清零以除能其它中断。

时基中断

使时基中断发生，INTC0 寄存器中总中断控制位 EMI、INTC1 寄存器中时基中断使能位 ETBI 必须先被置位。当时基中断发生时，INTC1 寄存器中 TBF 中断请求标志位将置位并触发时基中断。若中断使能，堆栈未满，当发生时基中断时，将调用位于地址 10H 处的子程序。当响应时基中断服务子程序时，中断请求标志位 TBF 会被复位且 EMI 位会被清零以除能其它中断。



时基中断

A/D 转换中断

使 A/D 转换中断发生，总中断控制位 EMI、A/D 转换中断使能位 EADI 必须先被置位。当 A/D 转换结束，INTC1 寄存器中 ADF 中断请求标志位将置位并触发 A/D 转换中断。若中断使能，堆栈未满，当发生 A/D 转换中断时，将调用位于地址 14H 处的子程序。当响应 AD 转换中断服务子程序时，中断请求标志位 ADF 会被复位且 EMI 位会被清零以禁止其它中断。

编程注意事项

通过禁止中断使能位，可以屏蔽中断请求，然而，一旦中断请求标志位被设定，它们会被保留在 INTC0 或 INTC1 中断控制寄存器内，直到相应的中断服务子程序执行或被软件指令清除。

建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断，当“CALL 子程序”在中断服务子程序中执行时，将破坏原来的控制序列。

所有中断都具有唤醒功能。当进入中断服务程序，系统仅将程序计数器的内容压入堆栈，如果有影响主程序流程的寄存器或状态寄存器被中断服务程序改变，应事先将这些数据保存起来。

复位和初始化

复位功能是任何单片机中基本的部分，使得单片机可以设定一些与外部参数无关的先置条件。最重要的复位条件是在初次提供电源给单片机后，经短暂延迟，内部电路将使得单片机被定义在良好的状态且准备执行第一条程序语句。上电复位之后，在程序未开始执行前，部分重要的内部寄存器将会被预先设定状态。程序计数器就是其中之一，它会被清除为零，使得单片机从最低的程序存储器地址开始执行程序。

除了上电复位外，即使单片机正在执行状态，有些情况的发生也迫使单片机必须加以复位。其中一个例子是当提供电源给单片机以执行程序后，$\overline{\text{RES}}$引脚被强制拉至低电平。这个例子为正常操作复位，单片机中只有一些寄存器受影响，而大部分寄存器不受影响，以便复位引脚回复至高电平后，单片机仍可以正常操作。复位的另一种形式是看门狗定时器溢出而复位单片机，所有复位操作类型导致不同的寄存器条件被加以设定。

另外一种复位为低电压复位，即 LVR 复位，在电源提供电压低于某一临界值的情况下，一种和$\overline{\text{RES}}$引脚复位类似的完全复位将会被执行。

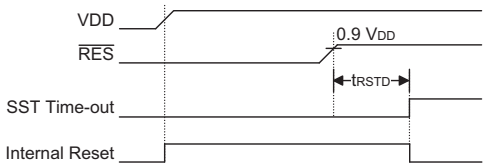
复位功能

通过内部与外部事件触发复位，单片机共有五种复位方式：

- 上电复位

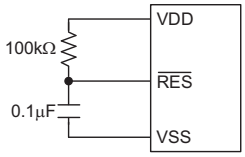
  这是最基本且不可避免的复位，发生在单片机上电后。除了保证程序存储器会从起始地址开始执行，上电复位也使得其它寄存器被设定在预设条件，所有的输入/输出端口寄存器和输入/输出端口控制寄存器在上电复位时会保持高电平，以确保上电后所有引脚被设为输入状态。

  虽然单片机有一个内部 RC 复位功能，如果电源上升缓慢或刚上电时电源不稳定，内部 RC 振荡可能会导致芯片复位不良，所以推荐使用和$\overline{\text{RES}}$引脚连接的外部 RC 电路。由 RC 电路所造成的时间延迟使得$\overline{\text{RES}}$引脚在电源供应稳定前的一段延长周期内保持在低电平。在这段时间内，单片机的正常操作是被禁止的。$\overline{\text{RES}}$引脚达到一定电压值后，再经过延迟时间 t_{RSTD}，单片机可开始进行正常操作。下图中 SST 是系统延迟周期 System Start-up Timer 的缩写。

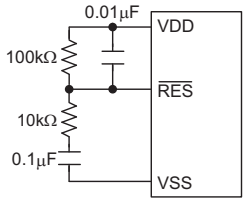


上电复位时序图

在许多应用场合，可以在 VDD 与$\overline{\text{RES}}$之间连接电阻，而在$\overline{\text{RES}}$与 VSS 之间连接电容作为复位电路，为了减少干扰影响，至$\overline{\text{RES}}$引脚的连线应尽量短。



基本复位电路



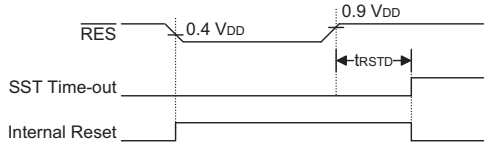
增强型复位电路

当系统在较强干扰的场合工作时，强烈建议使用增强型复位电路。

欲知更多外部复位电路的相关信息可参考 HOLTEK 网站应用范例 HA0075S。

- **RES**引脚复位

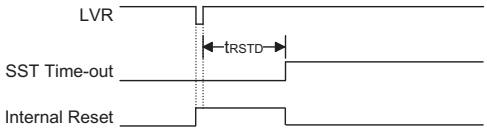
当单片机正常工作，RES引脚通过外部硬件(如外部开关)强迫拉至低电平时，此种复位形式即会发生。这种复位形式与其它复位一样，程序计数器会被清除为零且程序从头开始执行。



RES引脚复位时序图

- 低电压复位

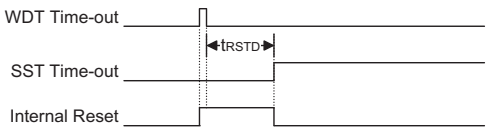
单片机具有低电压复位电路，用来监测它的电源电压，可通过掩膜选项进行选择。例如在更换电池的情况下，单片机供应的电压可能会落在 0.9V~V_{LVR} 的范围内，这时 LVR 将会自动复位单片机。LVR 包含以下的规格：有效的 LVR 信号，即在 0.9V~V_{LVR} 的低电压状态的时间，必须超过交流电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。V_{LVR} 参数值可通过掩膜选项进行设定。



低电压复位时序图

- 正常操作时看门狗溢出复位

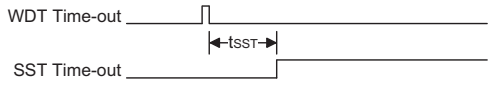
除了看门狗溢出标志位 TO 将被设为 1 之外，正常操作时看门狗溢出复位和RES复位相同。



正常操作时看门狗溢出复位时序图

- 暂停时看门狗溢出复位

  暂停时看门狗溢出复位和其它种类的复位有些不同，除了程序计数器与堆栈指针将被清 0 及 TO 位被设为 1 外，绝大部份的条件保持不变。图中 t_{SST} 的详细说明请参考交流电气特性。



暂停时看门狗溢出复位时序图

复位初始状态

不同的复位形式以不同的途径影响复位标志位。这些标志位，即 PDF 和 TO 位存放在状态寄存器中，由暂停功能或看门狗计数器等几种控制器操作控制。复位标志位如下所示：

TO	PDF	复位条件
0	0	上电时的 $\overline{RES}$ 复位
u	u	正常运行时的 $\overline{RES}$ 或 LVR 复位
1	u	正常运行时的 WDT 溢出复位
1	1	暂停时的 WDT 溢出复位

注：“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项 目	复位后情况
程序计数器	清除为零
中断	所有中断被禁止
看门狗定时器	WDT 清除并重新计时
定时/计数器	定时/计数器停止
预分频器	定时/计数器之预分频器内容清除
输入/输出口	所有 I/O 设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

不同的复位形式以不同的途径影响单片机中的内部寄存器。为保证复位发生后程序正常执行，了解单片机特定的复位后的情况是非常重要的。

下表描述了不同复位影响单片机的内部寄存器。

寄存器	复位 (上电复位)	$\overline{\text{RES}}$ 或 LVR 复位	WDT 溢出 (正常运行)	WDT 溢出 (暂停模式)
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--uu uuuu	--lu uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1000	0000 1000	0000 1000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PE	1111 1111	1111 1111	1111 1111	uuuu uuuu
PWM0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM2	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
INTC1	--00 --00	--00 --00	--00 --00	--uu --uu
LCDC	--00 0000	--00 0000	--00 0000	--uu uuuu
MODE	---0 ---0	---0 ---0	---0 ---0	---u ---u
ADRL	xxxx ----	xxxx ----	xxxx ----	uuuu ----
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR	0100 0000	0100 0000	0100 0000	uuuu uuuu
ACSR	10-- -000	10-- -000	10-- -000	1u-- -uuu

注： “u”表示未变化。
“x”表示未确定。
“-”表示未使用。

振荡器

不同的振荡器选择可以让使用者在不同的应用需求中获得更多范围的功能。有三种系统时钟可供选择，而看门狗定时器又有多种时钟源选项，提供了使用者最大的灵活性。

有三种方法产生系统时钟：

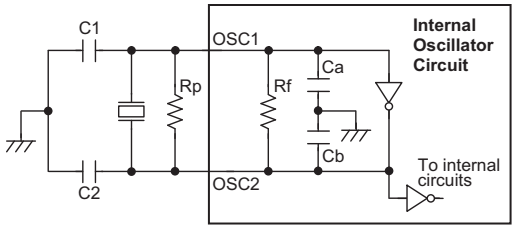
- 外部晶体/陶瓷振荡器
- 外部 RC 振荡器
- 外部 32768Hz 振荡器

当 MODE 寄存器中 MODS 位设置 32768Hz 晶体振荡器，外部晶体/陶瓷振荡器或 RC 系统振荡器可以通过掩膜选项进行设置。无论系统选择外部晶体/陶瓷振荡器或选择 RC 系统振荡器，32768Hz 晶体振荡器都必须连接。

欲知更多振荡器电路的相关信息可参考 HOLTEK 网站应用范例 HA0075S。

外部晶体/陶瓷振荡器

对于晶体振荡器的结构，只要简单地将晶体连接至 OSC1 和 OSC2，则会产生所需的相移及反馈，而不需其它外部的器件。然而为了保证某些低频率的晶体振荡和所有陶瓷共振器的应用，建议使用两个小容量电容 C1 和 C2，具体数值与客户选择的晶体/陶瓷晶振有关。外部并联的反馈电阻，即 R_P 在某些场合并不需要，仅用来辅助系统启动。



Note: 1. Rp is normally not required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

外部晶体/陶瓷振荡器

内部 Ca，Cb，Rf 典型值@5V，25℃		
Ca	Cb	Rf
11pF~13pF	13pF~15pF	470kΩ

振荡器内部元件值

晶体振荡器 C1 和 C2 值			
晶体频率	C1	C2	CL
12MHz	TBD	TBD	TBD
8MHz	TBD	TBD	TBD
4MHz	TBD	TBD	TBD
1MHz	TBD	TBD	TBD
注：1、C1，C2 值仅作为参考。 2、CL 为晶体规格测试时的负载电容。			

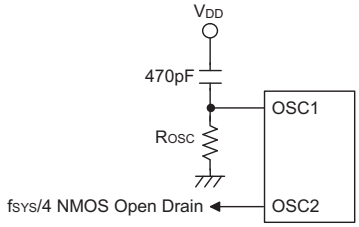
晶体振荡器电容推荐值

陶瓷振荡器 C1 和 C2 值		
陶瓷振荡频率	C1	C2
3.58MHz	TBD	TBD
1MHz	TBD	TBD
455KHz	TBD	TBD
注：C1，C2 值仅作为参考。		

陶瓷振荡器电容推荐值

外部 RC 振荡器

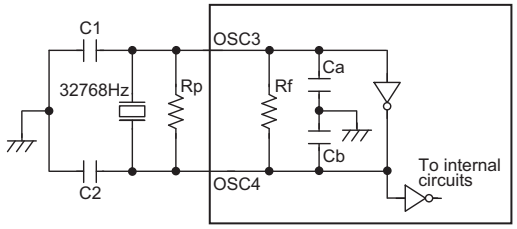
使用外部 RC 电路作为系统振荡器，需要在 OSC1 和 VDD 之间连接一个阻值约在 24kΩ到 1MΩ之间的电阻，OSC1 与 VSS 之间连接一个电容。虽然此振荡器配置成本较低，但振荡频率会因 VDD、温度和芯片本身的制成而改变，因此不适合用来做计时严格或需要精确振荡器频率的场合。对于外部电阻 R_{OSC} 的阻值，请参考附录章节中典型 RC 振荡器对温度以及对 V_{DD} 特性曲线分析。外部 RC 振荡器仅使用 OSC1 引脚，OSC1 与 PA6 引脚共用，留下 PA5 作为普通输入/输出使用。注意的是系统频率由内部电路和外部电阻 R_{OSC} 共同决定，图中显示的外部电容并不会影响振荡器的频率值。



外部 RC 振荡器

外部 RTC 振荡器

当系统进入暂停模式，系统时钟关闭以降低功耗。然而在某些应用，比如暂停模式下保持定时/计数功能，必须提供 32768Hz 振荡器的实时时钟。此时 OSC3 和 OSC4 引脚必须连接 32768Hz 晶体振荡器。为了保证低频率晶体振荡的应用，建议使用两个小容量电容 C1 和 C2，具体数值与客户选择的晶体规格有关。外部并联的反馈电阻，即 R_P 在某些场合并不需要，仅用来辅助系统启动。MODE 寄存器中 MODS 位用来选择外部 32768Hz 晶体振荡器或外部晶体/RC 作为系统时钟。系统使用 32768Hz 晶体振荡器时的慢速模式可降低功耗。



Note: 1. Rp is normally not required.
2. Although not shown OSC3/OSC4 pins have a parasitic capacitance of around 7pF.

内部 RC 振荡+外部 RTC 振荡

内部 Ca，Cb，Rf 典型值@5V，25℃		
Ca	Cb	Rf
TBD	TBD	TBD

RTC 振荡器内部元件值

RTC 振荡器 C1 和 C2 值			
晶体频率	C1	C2	CL
32768Hz	TBD	TBD	TBD
注：1、C1，C2 值仅作为参考。 2、CL 为晶体规格测试时的负载电容。			

32768Hz 晶体振荡器电容推荐值

当系统进入暂停模式，32768Hz 振荡器继续运行，如果 32768Hz 振荡器作为时钟和看门狗时钟源，将保持计数功能。

系统上电后，RTC 振荡器启动会有一定的延时。可以通过设置 MODE 寄存器中 QOSC 位缩短延时来控制快速振荡。系统上电时 QOSC 位清 0 以快速启动 RTC 振荡器，为了降低快速启动的功耗，建议系统上电 2 秒钟后，应用程序设置 QOSC 位为 1。注意的是，无论 QOSC 位是否设置，RTC 振荡器将保持正常工作，仅影响快速启动时的系统功耗。

看门狗定时振荡器

WDT 振荡器是一个完全独立且自由动作的内建振荡器，它在 5V 时的周期时间典型值为 65μs 且不需外部的器件搭配。当芯片进入暂停模式，系统时钟停止振荡，但 WDT 振荡器仍继续工作。然而在某些应用中，为了降低功耗，WDT 振荡器可以通过掩膜选项来关闭。

暂停模式下的暂停和唤醒

暂停模式

所有 HOLTEK 单片机都具有暂停功能。当单片机进入此模式，由于系统停止振荡，芯片的工作电流会降到极低静态模式。由于单片机保存了当前工作的一些内部条件，它可以被唤醒并继续运行，而不需要重新进行复位。这个特性对于电池供电系统等供电容量受限但需要单片机保存当前工作状态的应用场合特别重要。

进入暂停模式

单片机进入暂停模式仅能通过应用程序执行“HALT”指令实现，且造成如下结果：

- 系统振荡器将被关闭，应用程序将停止在“HALT”指令处
- 在 RAM 芯片和寄存器上的内容保持不变
- 如果 WDT 时钟源是来自 WDT 振荡器，则 WDT 将被清除然后再重新计数；若来源于系统时钟，则停止计数
- 所有输入/输出端口状态保持不变
- STATUS 寄存器中 PDF 标志位被置位，TO 标志位被清零

静态电流

要使系统静态电流降到最小，为毫安级，除了需要单片机进入 HALT 模式，还要考虑到电路的设计。特别要注意输入/输出口的状态。所有高阻抗输入引脚必须接高电平或低电平，否则会造成引脚浮空而引起内部振荡。另外还需要注意单片机输出端口上的负载，当外部电路为 CMOS 输入时，可设置为拉电流。若内部看门狗振荡器使能，也将消耗部分额外的电流。

唤醒

当系统进入 HALT 模式下，可以通过以下几种方式唤醒：

- 外部复位
- PA 口下降沿
- 系统中断
- WDT 溢出

若由外部 RES 引脚唤醒，系统会经过完全复位的过程。若由 WDT 溢出唤醒，则看门狗计数器将被复位清零。这两种唤醒方式都会使系统复位，可以通过状态寄存器中 TO 和 PDF 位来判断它的唤醒源。系统上电或执行清除看门狗的指令，PDF 被清零；执行 HALT 指令，PDF 将被置位。看门狗计数器溢出将会置位 TO 标志并唤醒系统，同时复位程序计数器和堆栈指针，其它标志保持原有状态。

端口 PA 中的每个位都可以通过掩膜选项独立选择唤醒功能。PA 端口唤醒后，程序将在“HALT”指令后继续执行。

如果系统是通过中断唤醒，则有两种可能发生，假如中断除能或中断使能但堆栈已满，程序将在“HALT”指令下一条处继续执行，但相应的中断服务程序只有在中断使能或堆栈空闲后被执行；假如中断使能且堆栈未满，则正常的中断响应将会发生。假设在进入暂停模式之前外部中断请求标志位被设为“1”，则相关中断的唤醒功能将无效。

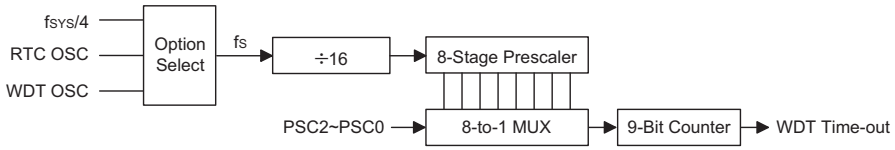
一旦唤醒事件发生，单片机回到正常运行将需要 1024 个系统时钟周期。如果唤醒由中断响应，则实际的中断子程序执行将延迟一个或数个周期；如果唤醒后接着去执行下一条指令，则它在 1024 个系统周期结束后立刻执行。

看门狗计数器

看门狗定时器的功能在于防止如电磁的干扰等外部不可控制事件，所造成的程序不正常动作或跳转到未知的地址。当 WDT 溢出时，它产生一个芯片复位的动作。WDT 时钟源可以通过掩膜选项进行设置，选择三个时钟源中一个提供，内建 WDT 振荡器，$f_{SYS}/4$ 或 RTC 振荡器。注意的是，假如 WDT 掩膜选项设为除能，则任何相关的指令将无效。

在此单片机中，所有看门狗定时器的选项，如使能/除能、WDT 时钟源和清除看门狗指令条数都是通过掩膜选项来选择，没有与 WDT 相关的内部寄存器。WDT 的时钟源之一是内部振荡器，在供应电压为 5V 时周期近似为 65μs。必须注意的是，这个内部时钟的周期可以随着 VDD、温度和制作工艺而改变。看门狗溢出周期为 $2^{14}/f_S \sim 2^{21}/f_S$。

如果选择 $f_{SYS}/4$ 为看门狗时钟源，必须注意的是，当系统进入暂停模式后，指令时钟会停止且 WDT 将失去其保护目的。在这种情况下，系统只能通过外部逻辑重新复位。当系统操作在干扰严重的环境时，建议使用内部 WDT 振荡器。



看门狗定时器

系统在正常运行状态下，WDT 溢出将导致芯片复位，并置位状态标志位 TO。然而如果系统处于暂停模式，则只有一个从暂停模式来的 WDT 溢出复位发生，它只复位程序计数器和 SP。有三种方法可以用来清除 WDT 的内容，第一种是外部硬件复位($\overline{RES}$引脚低电平)，第二种是通过软件指令，而第三种是通过“HALT”指令。

使用软件指令有两种方法去清除看门狗寄存器，必须由掩膜选项选择。第一种选择是使用单一“CLR WDT”指令，而第二种是使用“CLR WDT1”和“CLR WDT2”两个指令。对于第一种选择，只要执行“CLR WDT”便清除 WDT。而第二种选择，必须交替执行“CLR WDT1”和“CLR WDT2”两者才能成功的清除 WDT。关于第二种选择要注意的是，如果“CLR WDT1”正被使用来清除 WDT，接着再执行这条指令将是无效的，只有执行“CLR WDT2”指令才能清除 WDT。同样的“CLR WDT2”指令已经执行后，只有接着执行“CLR WDT1”指令才可以清除看门狗定时器。

蜂鸣器

和可编程分频器类似，蜂鸣器提供了一种精确频率输出的方法，适用于压电蜂鸣器和外部电路需要精确频率的场合。蜂鸣器输出 BZ 和$\overline{BZ}$与 PA0 和 PA1 共用引脚，可通过掩膜选项设置蜂鸣器的功能，三种工作方式为 PA0 和 PA1 作为正常输入/输出使用，PA0 和 PA1 作为 BZ 和$\overline{BZ}$功能，以及 PA0 作为 BZ 蜂鸣器输出功能，而 PA1 作为正常输入/输出使用。注意的是，$\overline{BZ}$作为 BZ 的取反输出，蜂鸣器的差分输出用来增加输出功率。

蜂鸣器时钟源为系统内部时钟 f_S，经掩膜选项设置分频系数，由此来控制蜂鸣器输出的频率范围为 $f_S/2 \sim f_S/4$。系统时钟来源于 RTC 振荡器，WDT 振荡器或系统时钟 4 分频，由 f_S 时钟源掩膜选项进行设置。注意的是，如果选择 RTC 振荡器为系统时钟，则 f_S 和对应的蜂鸣器输出必须选择 RTC 振荡器为时钟源。蜂鸣器频率由内部 f_S 时钟源和内部分频系数掩膜选项控制，单片机内部并无相关寄存器影响蜂鸣器。

如果掩膜选项设置 PA0 和 PA1 为蜂鸣器输出 BZ 和$\overline{BZ}$功能，为了保证蜂鸣器正常工作，PAC 端口控制寄存器中 PAC0 和 PAC1 必须设置为输出，PA 数据寄存器中 PA0 必须设置为 1，以使能蜂鸣器输出，如果 PA0 设置为 0，则 PA0 和 PA1 保持为低电平。换句话说，PA 寄存器中 PA0 为蜂鸣器输出 BZ 和$\overline{BZ}$功能的开关。注意的是，PA 寄存器中 PA1 并非控制蜂鸣器输出$\overline{BZ}$。

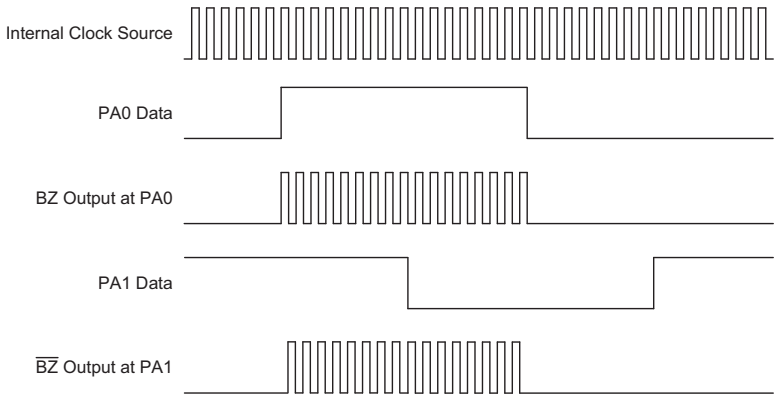
PA0/PA1 输入/输出功能：

PAC 寄存器 PAC0	PAC 寄存器 PAC1	PA 数据寄存器 PA0	PA 数据寄存器 PA1	输出功能
0	0	1	X	PA0=BZ，PA1=$\overline{BZ}$
0	0	0	X	PA0=0，PA1=0
0	1	1	X	PA0=BZ，PA1=Input
0	1	0	X	PA0=0，PA1=Input
1	0	X	D	PA0= Input ，PA1=D
1	1	X	X	PA0= Input，PA1=Input

注： “X”表示任意
“D”表示数据“0”或“1”

一旦掩膜选项设置 PA0 为 BZ 蜂鸣器功能，PA1 将作为正常输入/输出使用。PA0 作为 BZ 蜂鸣器输出功能时，PAC 控制寄存器中 PAC0 必须设置为 0，同时 PA 数据寄存器中 PA0 必须设置为 1，以使能蜂鸣器输出，如果 PA0 设置为 0，则 PA0 保持为低电平。换句话说，PA 寄存器中 PA0 为蜂鸣器输出 BZ 功能的开关。注意的是，PA 寄存器中 PA1 并非控制蜂鸣器输出。如果将 PAC 寄存器中 PAC0 设置为 1，即使掩膜选项设置了 BZ 蜂鸣器功能，PA0 仍作为输入使用。

注意的是，无论掩膜选项中如何设置蜂鸣器功能，一旦端口控制寄存器设置为输入，系统将忽略掩膜选项设置，总是作为输入引脚使用。如此设计可以将引脚作为蜂鸣器输出和输入功能，而无须关心掩膜选项的设置，相应引脚的功能可在应用程序的控制下修改端口控制寄存器的方法动态地实现。



蜂鸣器输出引脚控制

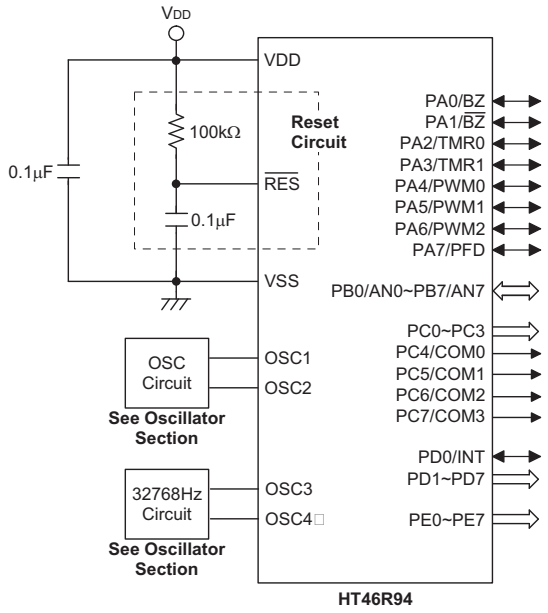
注：上述图表的条件是在掩膜选项设置 PA0 和 PA1 为 BZ 和$\overline{BZ}$蜂鸣器输出。端口控制寄存器已经设置为输出。PA1 的数据设置并不影响蜂鸣器输出。

掩膜选项

掩膜选项在烧写程序时写入芯片。通过 HT-IDE 的软件开发环境，使用者在开发过程中可以选择掩膜选项。当掩膜选项烧入单片机后，无法再通过应用程序修改。所有位必须按系统的需要定义，具体内容可参考下表：

编号	选项
输入/输出选项	
1	PA0~PA7 位唤醒：使能/禁止（位选择）
2	PA 选择上拉电阻：使能/禁止（端口选择）
振荡器选项	
3	系统振荡器：晶体或 RC
PWM 选项	
4	PA4~PA6：PWM0~PWM2 功能选择
5	PWM 模式：6+2 或 7+1 模式
定时器选项	
6	定时/计数器 0 时钟源：$f_{SYS}/4$ 或 f_{SP}
PFD 选项	
7	PA7：正常输入/输出或 PFD 输出
蜂鸣器选项	
8	蜂鸣器功能：单个 BZ 使能，BZ 和 $\overline{BZ}$ 都使能，BZ 和 $\overline{BZ}$ 都禁止
9	蜂鸣器频率：$f_S/2$ ，$f_S/4$ ，$f_S/8$， $f_S/16$
时基选项	
10	时基溢出周期：$f_S/2^5$ ，$f_S/2^6$ ，$f_S/2^7$ ，$f_S/2^8$ ，$f_S/2^9$ ，$f_S/2^{10}$ ，$f_S/2^{11}$ ，$f_S/2^{12}$
看门狗选项	
11	看门狗时钟源：WDT 振荡器，RTC 振荡器或 $f_{SYS}/4$
12	看门狗功能：使能/禁止
13	清除看门狗指令：1 条或 2 条
LVR 选项	
14	LVR 功能：使能/禁止
15	LVR 电压：2.1V，3.15V 或 4.2V
LOCK 选项	
16	全部 LOCK
17	局部 LOCK

应用电路


VDD
VDD
100kΩ
Reset
Circuit
0.1μF
RES
0.1μF
VSS
PA0/BZ
PA1/BZ
PA2/TMR0
PA3/TMR1
PA4/PWM0
PA5/PWM1
PA6/PWM2
PA7/PFD
PB0/AN0~PB7/AN7
PC0~PC3
PC4/COM0
PC5/COM1
PC6/COM2
PC7/COM3
OSC
Circuit
OSC1
OSC2
See Oscillator
Section
PD0/INT
PD1~PD7
32768Hz
Circuit
OSC3
OSC4
PE0~PE7
See Oscillator
Section
HT46R94

指令集

简介

任何单片机成功运作的核心在于它的指令集，此指令集为一组程序指令码，用来指导单片机如何去执行指定的工作。在盛群单片机中，提供了丰富且灵活的指令，共超过六十条，程序设计者可以事半功倍地实现他们的应用。

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

大部分的操作均只需要一个指令周期来执行。分支、调用或查表则需要两个指令周期。一个指令周期相当于四个系统时钟周期，因此如果在 8MHz 的系统时钟振荡器下，大部分的操作将在 0.5μs 中执行完成，而分支或调用操作则将在 1μs 中执行完成。虽然需要两个指令周期的指令通常指的是 JMP、CALL、RET、RETI 和查表指令，但如果牵涉到程序计数器低字节寄存器 PCL 也将多花费一个周期去加以执行。即指令改变 PCL 的内容进而导致直接跳转至新地址时，需要多一个周期去执行，例如“CLR PCL”或“MOV PCL, A”指令。对于跳转指令必须注意的是，如果比较的结果牵涉到跳转动作将多花费一个周期，如果没有则需一个周期即可。

数据的传送

单片机程序中数据传送是使用最为频繁的操作之一，使用三种 MOV 的指令，数据不但可以从寄存器转移至累加器(反之亦然)，而且能够直接移动立即数到累加器。数据传送最重要的应用之一是从输入端口接收数据或者传送数据到输出端口。

算术运算

算术运算和数据处理是大部分单片机应用所必需具备的能力，在盛群单片机内部的指令集中，可直接实现加与减的运算。当加法的结果超出 255 或减法的结果少于 0 时，要注意正确的处理进位和借位的问题。INC、INCA、DEC 和 DECA 指令提供了对一个指定地址的值加一或减一的功能。

逻辑和移位运算

标准逻辑运算例如 AND、OR、XOR 和 CPL 全都包含在盛群单片机内部的指令集中。大多数牵涉到数据运算的指令，数据的传送必须通过累加器。在所有逻辑数据运算中，如果运算结果为零，则零标志位将被置位，另外逻辑数据运用形式还有移位指令，例如 RR、RL、RRC 和 RLC 提供了向左或向右移动一位的方法。不同的移位指令可满足不同的应用需要。移位指令常用于串行端口的程序应用，数据可从内部寄存器转移至进位标志位，而此位则可被检验，移位运算还可应用在乘法与除法的运算组成中。

分支和控制的转换

程序分支是采取使用 JMP 指令跳转至指定地址或使用 CALL 指令调用子程序的形式，两者之不同在于当子程序被执行完毕后，程序必须马上返回原来的地址。这个动作是由放置在子程序里的返回指令 RET 来实现，它可使程序跳回 CALL 指令之后的地址。在 JMP 指令中，程序则只是跳到一个指定的地址而已，并不需如 CALL 指令般跳回。一个非常有用的分支指令是条件跳转，跳转条件是由数据存储器或指定位来加以决定。遵循跳转条件，程序将继续执行下一条指令或略过且跳转至接下来的指令。这些分支指令是程序走向的关键，跳转条件可能是外部开关输入，或者是内部数据位的值。

位运算

提供数据存储器中单个位的运算指令是盛群单片机的特性之一。这特性对于输出端口位的设置尤其有用，其中个别的位或端口的引脚可以使用“SET [m].i”或“CLR [m].i”指令来设定其为高位或低位。如果没有这特性，程序设计师必须先读入输出口的 8 位数据，处理这些数据，然后再输出正确的新数据。这种读入-修改-写出的过程现在则被位运算指令所取代。

查表运算

数据的储存通常由寄存器完成，然而当处理大量固定的数据时，它的存储量常常造成对个别存储器的不便。为了改善此问题，盛群单片机允许在程序存储器中建立一个表格作为数据可直接存储的区域，只需要一组简易的指令即可对数据进行查表。

其它运算

除了上述功能指令外，其它指令还包括用于省电的“HALT”指令和使程序在极端电压或电磁环境下仍能正常工作的看门狗定时器控制指令。这些指令的使用则请查阅相关的章节。

指令集概要

惯例

x：立即数

m：数据存储器地址

A：累加器

i：第 0~7 位

addr：程序存储器地址

助记符	说明	指令周期	影响标志位
算术运算			
ADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	1	Z,C,AC,OV
ADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	1 注	Z,C,AC,OV
ADD A,x	ACC 与立即数相加，结果放入 ACC	1	Z,C,AC,OV
ADC A,[m]	ACC 与数据存储器、进位标志相加，结果放入 ACC	1	Z,C,AC,OV
ADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	1 注	Z,C,AC,OV
SUB A,x	ACC 与立即数相减，结果放入 ACC	1	Z,C,AC,OV
SUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	1	Z,C,AC,OV
SUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	1 注	Z,C,AC,OV
SBC A,[m]	ACC 与数据存储器、进位标志的反相减，结果放入 ACC	1	Z,C,AC,OV
SBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	1 注	Z,C,AC,OV
DAA [m]	将加法运算中放入 ACC 的值调整为十进制数，并将结果放入数据存储器	1 注	C
逻辑运算			
AND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	1	Z
OR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	1	Z
XOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	1	Z
ANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	1 注	Z
ORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	1 注	Z
XORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	1 注	Z
AND A,x	ACC 与立即数做“与”运算，结果放入 ACC	1	Z
OR A,x	ACC 与立即数做“或”运算，结果放入 ACC	1	Z
XOR A,x	ACC 与立即数做“异或”运算，结果放入 ACC	1	Z
CPL [m]	对数据存储器取反，结果放入数据存储器	1 注	Z
CPLA [m]	对数据存储器取反，结果放入 ACC	1	Z
递增和递减			
INCA [m]	递增数据存储器，结果放入 ACC	1	Z
INC [m]	递增数据存储器，结果放入数据存储器	1 注	Z
DECA [m]	递减数据存储器，结果放入 ACC	1	Z
DEC [m]	递减数据存储器，结果放入数据存储器	1 注	Z

助记符	说明	指令周期	影响标志位
移位			
RRA [m]	数据存储器右移一位，结果放入 ACC	1	无
RR [m]	数据存储器右移一位，结果放入数据存储器	1 注	无
RRCA [m]	带进位将数据存储器右移一位，结果放入 ACC	1	C
RRC [m]	带进位将数据存储器右移一位，结果放入数据存储器	1 注	C
RLA [m]	数据存储器左移一位，结果放入 ACC	1	无
RL [m]	数据存储器左移一位，结果放入数据存储器	1 注	无
RLCA [m]	带进位将数据存储器左移一位，结果放入 ACC	1	C
RLC [m]	带进位将数据存储器左移一位，结果放入数据存储器	1 注	C
数据传送			
MOV A,[m]	将数据存储器送至 ACC	1	无
MOV [m],A	将 ACC 送至数据存储器	1 注	无
MOV A,x	将立即数送至 ACC	1	无
位运算			
CLR [m].i	清除数据存储器的位	1 注	无
SET [m].i	置位数据存储器的位	1 注	无
转移			
JMP addr	无条件跳转	2	无
SZ [m]	如果数据存储器为零，则跳过下一条指令	1 注	无
SZA [m]	数据存储器送至 ACC，如果内容为零，则跳过下一条指令	1 注	无
SZ [m].i	如果数据存储器的第 i 位为零，则跳过下一条指令	1 注	无
SNZ [m].i	如果数据存储器的第 i 位不为零，则跳过下一条指令	1 注	无
SIZ [m]	递增数据存储器，如果结果为零，则跳过下一条指令	1 注	无
SDZ [m]	递减数据存储器，如果结果为零，则跳过下一条指令	1 注	无
SIZA [m]	递增数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令	1 注	无
SDZA [m]	递减数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令	1 注	无
CALL addr	子程序调用	2	无
RET	从子程序返回	2	无
RET A,x	从子程序返回，并将立即数放入 ACC	2	无
RETI	从中断返回	2	无
查表			
TABRDC [m]	读取当前页的 ROM 内容，并送至数据存储器和 TBLH	2 注	无
TABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	2 注	无
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1 注	无
SET [m]	置位数据存储器	1 注	无
CLR WDT	清除看门狗定时器	1	TO,PDF
CLR WDT1	预清除看门狗定时器	1	TO,PDF
CLR WDT2	预清除看门狗定时器	1	TO,PDF
SWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	1 注	无
SWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	1	无
HALT	进入暂停模式	1	TO,PDF

注： 1、对跳转指令而言，如果比较的结果牵涉到跳转即需 2 个周期，如果没有发生跳转，则只需一个周期。

2、任何指令若要改变 PCL 的内容将需要 2 个周期来执行。

3、对于“CLR WDT1”或“CLR WDT2”指令而言，TO 和 PDF 标志位也许会受执行结果影响，“CLR WDT1”和“CLR WDT2”被连续地执行后，TO 和 PDF 标志位会被清除，除此之外 TO 和 PDF 标志位保持不变。

指令定义

ADC A, [m] Add data memory and carry to the accumulator
说明： 将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
运算过程： ACC←ACC+[m]+C
影响标志位： OV、Z、AC、C

ADCM A, [m] Add the accumulator and carry to the accumulator
说明： 将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
运算过程： [m]←ACC+[m]+C
影响标志位： OV、Z、AC、C

ADD A, [m] Add data memory to the accumulator
说明： 将指定的数据存储器和累加器内容相加，结果存放到累加器。
运算过程： ACC←ACC+[m]
影响标志位： OV、Z、AC、C

ADD A, x Add immediate data to the accumulator
说明： 将累加器和立即数相加，结果存放到累加器。
运算过程： ACC←ACC+x
影响标志位： OV、Z、AC、C

ADDM A, [m] Add the accumulator to the data memory
说明： 将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
运算过程： [m]←ACC+[m]
影响标志位： OV、Z、AC、C

AND A, [m] Logical AND accumulator with data memory
说明： 将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。
运算过程： ACC←ACC “AND” [m]
影响标志位： Z

AND A, x Logical AND immediate data to the accumulator

说明： 将累加器中的数据和立即数做逻辑与，结果存放到累加器。
运算过程： ACC←ACC “AND” x
影响标志位： Z

ANDM A, [m] Logical AND data memory with the accumulator
说明： 将指定数据存储器内容和累加器中的数据做逻辑与，结果存放到数据存储器。
运算过程： [m]←ACC “AND” [m]
影响标志位： Z

CALL addr Subroutine call

说明： 无条件的调用指定地址的子程序，此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈，接着载入指定地址并从新地址执行程序。由于指令需要额外的运算，所以此指令为 2 个周期。

运算过程： Stack←Program Counter+1
Program Counter ← addr

影响标志位： 无

CLR [m] Clear data memory

说明： 将指定数据存储器的内容清零。

运算过程： [m] ←00H

影响标志位： 无

CLR [m] . i Clear bit of data memory

说明： 将指定数据存储器的 i 位内容清零。

运算过程： [m].i ←0

影响标志位： 无

CLR WDT Clear Watchdog Timer

说明： WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。

运算过程： WDT ← 00H
PDF & TO ← 0

影响标志位： TO、PDF

CLR WDT1 Preclear Watchdog Timer

说明： PDF 和 TO 标志位都被清 0。必须配合 CLR WDT2 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT1，而没有执行 CLR WDT2 时，PDF 与 TO 保留原状态不变。

运算过程： WDT ← 00H
PDF & TO ← 0

影响标志位： TO、PDF

CLR WDT2 Preclear Watchdog Timer

说明： PDF 和 TO 标志位都被清 0。必须配合 CLR WDT1 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT2，而没有执行 CLR WDT1 时，PDF 与 TO 保留原状态不变。

运算过程： WDT ← 00H
PDF & TO ← 0

影响标志位： TO、PDF

CPL [m] Complement data memory

说明： 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或从 0 变 1。

运算过程： [m] ← [$\overline{m}$]

影响标志位： Z

CPLA [m] Complement data memory
说明： 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或从 0 变 1，结果被存放回累加器且数据寄存器的内容保持不变。
运算过程： ACC ← [$\overline{m}$]
影响标志位： Z

DAA [m] Decimal-Adjust accumulator for addition
说明： 将累加器中的内容转换为 BCD（二进制转成十进制）码。如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执行对原值加“6”，否则原值保持不变；如果高四位的值大于“9”或 C=1，那么 BCD 调整就执行对原值加“6”。BCD 转换实质上是根据累加器和标志位执行 00H，06H，60H 或 66H 的加法运算，结果存放到数据存储器。只有进位标志位 C 受影响，用来指示原始 BCD 的和是否大于 100，并可以进行双精度十进制数的加法运算。
操作： [m] ←ACC+00H 或 [m] ←ACC+06H
[m] ←ACC+60H 或 [m] ←ACC+66H
影响标志位： C

DEC [m] Decrement data memory
说明： 将指定数据存储器的内容减 1。
运算过程： [m] ← [m]-1
影响标志位： Z

DECA [m] Decrement data memory and place result in the accumulator
说明： 将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。
运算过程： ACC ← [m]-1
影响标志位： Z

HALT Enter power down mode
说明： 此指令终止程序执行并关掉系统时钟，RAM 和寄存器的内容保持原状态，WDT 计数器和分频器被清“0”，暂停标志位 PDF 被置位 1，WDT 溢出标志位 TO 被清 0。
运算过程： PDF ← 1
TO ← 0
影响标志位： TO、PDF

INC [m] Increment data memory

说明： 将指定数据存储器的内容加 1。
运算过程： [m] ← [m]+1
影响标志位： Z

INCA [m] Increment data memory and place result in the accumulator
说明： 将指定数据存储器的内容加 1，结果存放回累加器并保持指定的数据存储器内容不变。
运算过程： ACC ← [m]+1
影响标志位： Z

JMP addr Directly jump
说明： 程序计数器的内容无条件地由被指定的地址取代，程序由新的地址继续执行。当新的地址被加载时，必须插入一个空指令周期，所以此指令为 2 个周期的指令。
运算过程： PC ← addr
影响标志位： 无

MOV A, [m] Move data memory to the accumulator
说明： 将指定数据存储器的内容复制到累加器。
运算过程： ACC ← [m]
影响标志位： 无

MOV A, x Move immediate data to the accumulator
说明： 将 8 位立即数载入累加器。
运算过程： ACC ← x
影响标志位： 无

MOV [m], A Move the accumulator data to memory
说明： 将累加器的内容复制到指定的数据存储器。
运算过程： [m] ← ACC
影响标志位： 无

NOP No operation
说明： 空操作，顺序执行下一条指令。
运算过程： PC ← PC+1
影响标志位： 无

OR A, [m] Logical OR accumulator with data memory
说明： 将累加器中的数据和指定的数据存储器内容逻辑或，结果存放到累加器。
运算过程： ACC←ACC “OR” [m]
影响标志位： Z

OR A, x Logical OR immediate data to the accumulator
说明： 将累加器中的数据和立即数逻辑或，结果存放到累加器。
运算过程： ACC←ACC “OR” x
影响标志位： Z

ORM A, [m] Logical OR data memory with accumulator
说明： 将存在指定数据存储器中的数据和累加器逻辑或，结果放到数据存储器。
运算过程： [m]←ACC “OR” [m]
影响标志位： Z

RET Return from subroutine
说明： 将堆栈寄存器中的程序计数器值恢复，程序由取回的地址继续执行。
运算过程： PC ← Stack
影响标志位： 无

RET A, x Return and place immediate data in the accumulator

说明： 将堆栈寄存器中的程序计数器值恢复且累加器载入指定的立即数，程序由取回的地址继续执行。

运算过程： PC ← Stack
ACC ← x

影响标志位： 无

RETI Return from interrupt

说明： 将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被相应，则这个中断将在返回主程序之前被相应。

运算过程： PC ← Stack
EMI ← 1

影响标志位： 无

RL [m] Rotate data memory left

说明： 将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。

运算过程： [m].(i+1) ←[m].i (i=0~6)
[m].0 ←[m].7

影响标志位： 无

RLA [m] Rotate data memory left and place result in the accumulator

说明： 将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。

运算过程： ACC.(i+1) ←[m].i (i=0~6)
ACC.0 ←[m].7

影响标志位： 无

RLC [m] Rotate data memory left through carry

说明： 将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位。

运算过程： [m].(i+1) ←[m].i (i=0~6)
[m].0 ←C
C ← [m].7

影响标志位： C

RLCA [m] Rotate left through carry and place result in the accumulator

说明： 将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位,移位结果送回累加器，但是指定数据寄存器的内容保持不变。

运算过程： ACC.(i+1) ←[m].i (i=0~6)
ACC.0 ←C
C ← [m].7

影响标志位： C

RR [m] Rotate data memory right

说明： 将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。

运算过程： [m].i ←[m].(i+1) (i=0~6)
[m].7 ←[m].0,

影响标志位： 无

RRA [m] Rotate right and place result in the accumulator

说明： 将指定数据存储器的内容循环右移 1 位，第 0 位移到第 7 位，移位结果存放到累加器，而指定数据存储器的内容保持不变。

运算过程： ACC.i ←[m].(i+1) (i=0~6)
ACC.7 ←[m].0

影响标志位： 无

RRC [m] Rotate data memory right through carry

说明： 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位。

运算过程： [m].i ←[m].(i+1) (i=0~6)
[m]. 7 ←C
C ← [m].0

影响标志位： C

RRCA [m] Rotate right through carry and place result in the accumulator

说明： 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位,移位结果送回累加器，但是指定数据寄存器的内容保持不变。

运算过程： ACC.i ←[m].(i+1) (i=0~6)
ACC.7 ←C
C ← [m].0

影响标志位： C

SBC A,[m] Subtract data memory and carry from the accumulator

说明： 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

运算过程： ACC←ACC－[m]－$\overline{C}$

影响标志位： OV、Z、AC、C

SBCM A,[m] Subtract data memory and carry from the accumulator

说明： 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

运算过程： ACC←ACC－[m]－$\overline{C}$

影响标志位： OV、Z、AC、C

SDZ [m] Skip if decrement data memory is 0

说明： 将指定的数据存储器的内容减 1，判断是否为 0，若为 0 则跳过下一条指令，由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

运算过程： [m] ← [m] -1，如果[m]=0 跳过下一条指令执行

影响标志位： 无

SDZA [m] Decrement data memory and place result in ACC,skip if 0

说明： 将指定数据存储器内容减 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

运算过程： ACC ← [m]-1，如果 ACC=0 跳过下一条指令执行

影响标志位： 无

SET [m] Set data memory

说明： 将指定数据存储器的每一位设置为 1。

运算过程： [m] ← FFH

影响标志位： 无

SET [m]. i Set bit of data memory

说明： 将指定数据存储器的第 i 位设置为 1。

运算过程： [m].i ←1

影响标志位： 无

SIZ [m] Skip if increment data memory is 0

说明： 将指定的数据存储器的内容加 1，判断是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

运算过程： [m] ←[m]+1，如果[m]=0 跳过下一条指令执行

影响标志位： 无

SIZA [m] Increment data memory and place result in ACC,skip if 0

说明： 将指定数据存储器的内容加 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果会被存放到累加器，但是指定数据存储器的内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

运算过程： ACC ←[m]+1，如果 ACC=0 跳过下一条指令执行

影响标志位： 无

SNZ [m]. i Skip if bit I of the data memory is not 0

说明： 判断指定数据存储器的第 i 位，若不为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果为 0，则程序继续执行下一条指令。

运算过程： 如果[m].i≠0，跳过下一条指令执行

影响标志位： 无

SUB A, [m] Subtract data memory from the accumulator
说明： 将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
运算过程： ACC←ACC－[m]
影响标志位： OV、Z、AC、C

SUBM A, [m] Subtract data memory from the accumulator
说明： 将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
运算过程： [m]←ACC－[m]
影响标志位： OV、Z、AC、C

SUB A, x Subtract immediate data from the accumulator
说明： 将累加器的内容减去立即数，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
运算过程： ACC←ACC－x
影响标志位： OV、Z、AC、C

SWAP [m] Swap nibbles within the data memory
说明： 将指定数据存储器的低 4 位和高 4 位互相交换。
运算过程： [m].3~[m].0 ↔[m].7~[m].4
影响标志位： 无

SWAPA [m] Swap data memory and place result in the accumulator
说明： 将指定数据存储器的低 4 位和高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。
运算过程： ACC.3~ACC.0← [m].7~[m].4
ACC.7~ACC.4← [m].3~[m].0
影响标志位： 无

SZ [m] Skip if data memory is 0
说明： 判断指定数据存储器的内容是否为 0，若为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
运算过程： 如果[m] = 0, 跳过下一条指令执行
影响标志位： 无

SZA [m] Move data memory to ACC,skip if 0
说明： 将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
运算过程： ACC←[m]，如果[m] = 0，跳过下一条指令执行
影响标志位： 无

SZ [m]. i Skip if bit I of the data memory is 0
说明： 判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
运算过程： 如果[m].i = 0，跳过下一条指令执行
影响标志位： 无

TABRDC [m] Move the ROM code(current page) to TBLH and data memory
说明： 将表格指针 TBLP 所指的程序代码低字节（当前页）移至指定的数据存储器且将高字节移至 TBLH。
运算过程： [m] ←程序代码（低字节）
TBLH←程序代码（高字节）
影响标志位： 无

TABRDL [m] Move the ROM code(last page) to TBLH and data memory
说明： 将表格指针 TBLP 所指的程序代码低字节（最后一页）移至指定的数据存储器且将高字节移至 TBLH。
运算过程： [m] ←程序代码（低字节）
TBLH←程序代码（高字节）
影响标志位： 无

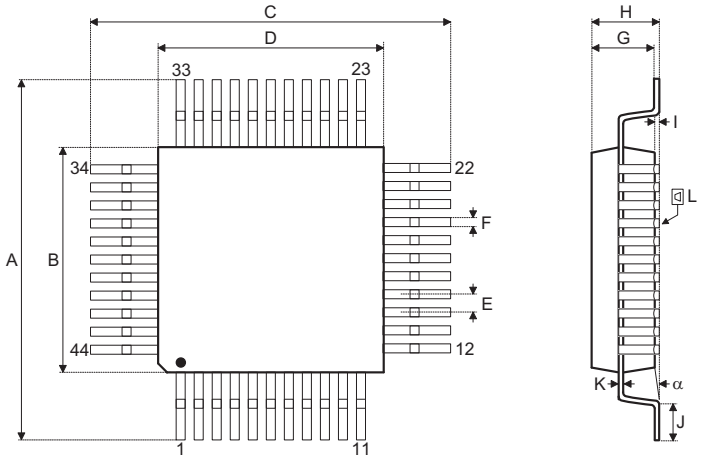
XOR A, [m] Logical XOR accumulator with data memory
说明： 将累加器的数据和指定的数据存储器内容逻辑异或，结果存放到累加器。
运算过程： ACC←ACC “XOR” [m]
影响标志位： Z

XORM A, [m] Logical XOR data memory with accumulator
说明： 将累加器的数据和指定的数据存储器内容逻辑异或，结果放到数据存储器。
运算过程： [m]←ACC “XOR” [m]
影响标志位： Z

XOR A, x Logical XOR immediate data to the accumulator
说明： 将累加器的数据与立即数逻辑异或，结果存放到累加器。
运算过程： ACC←ACC “XOR” x
影响标志位： Z

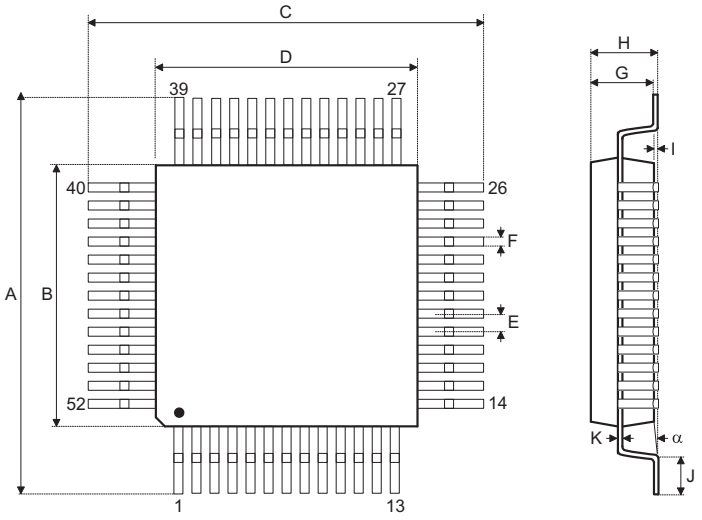
封装尺寸

44-pin QFP (10×10) 外形尺寸



符号	尺寸 (单位：mm)		
	最小	一般	最大
A	13	—	13.4
B	9.9	—	10.1
C	13	—	13.4
D	9.9	—	10.1
E	—	0.8	—
F	—	0.3	—
G	1.9	—	2.2
H	—	—	2.7
I	0.25	—	0.5
J	0.73	—	0.93
K	0.1	—	0.2
L	—	0.1	—
α	0°	—	7°

52-pin QFP (14×14) 外形尺寸



符号	尺寸 (单位：mm)		
	最小	一般	最大
A	17.3	—	17.5
B	13.9	—	14.1
C	17.3	—	17.5
D	13.9	—	14.1
E	—	1	—
F	—	0.4	—
G	2.5	—	3.1
H	—	—	3.4
I	—	0.1	—
J	0.73	—	1.03
K	0.1	—	0.2
α	0°	—	7°

盛群半导体股份有限公司（总公司）
新竹市科学工业园区研新二路 3 号
电话: 886-3-563-1999
传真: 886-3-563-1189
网站: www.holtek.com.tw

盛群半导体股份有限公司（台北业务处）
台北市南港区园区街 3 之 2 号 4 楼之 2
电话: 886-2-2655-7070
传真: 886-2-2655-7373
传真: 886-2-2655-7383 (International sales hotline)

盛扬半导体有限公司（上海业务处）
上海宜山路 2016 号合川大厦 1 号楼 3 楼 G 室 201103
电话: 021-5422-4590
传真: 021-5422-4605
网站: www.holtek.com.cn

盛扬半导体有限公司（深圳业务处）
深圳市南山区科技园科技中三路与高新中二道交汇处生产力大楼 A 单元五楼 518057
电话: 0755-8616-9908,8616-9308
传真: 86-755-8616-9722

盛扬半导体有限公司（北京业务处）
北京市西城区宣武门西大街甲 129 号金隅大厦 1721 室 100031
电话: 010-6641-0030, 6641-7751, 6641-7752
传真: 010-6641-0125

盛扬半导体有限公司（成都业务处）
成都市东大街 97 号香槟广场 C 座 709 室 610016
电话: 028-6653-6590
传真: 028-6653-6591

Holmate Semiconductor, Inc.（北美业务处）
46712 Fremont Blvd., Fremont, CA 94538
电话: 510-252-9880
传真: 510-252-9885
网站: http://www.holtek.com