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

特性

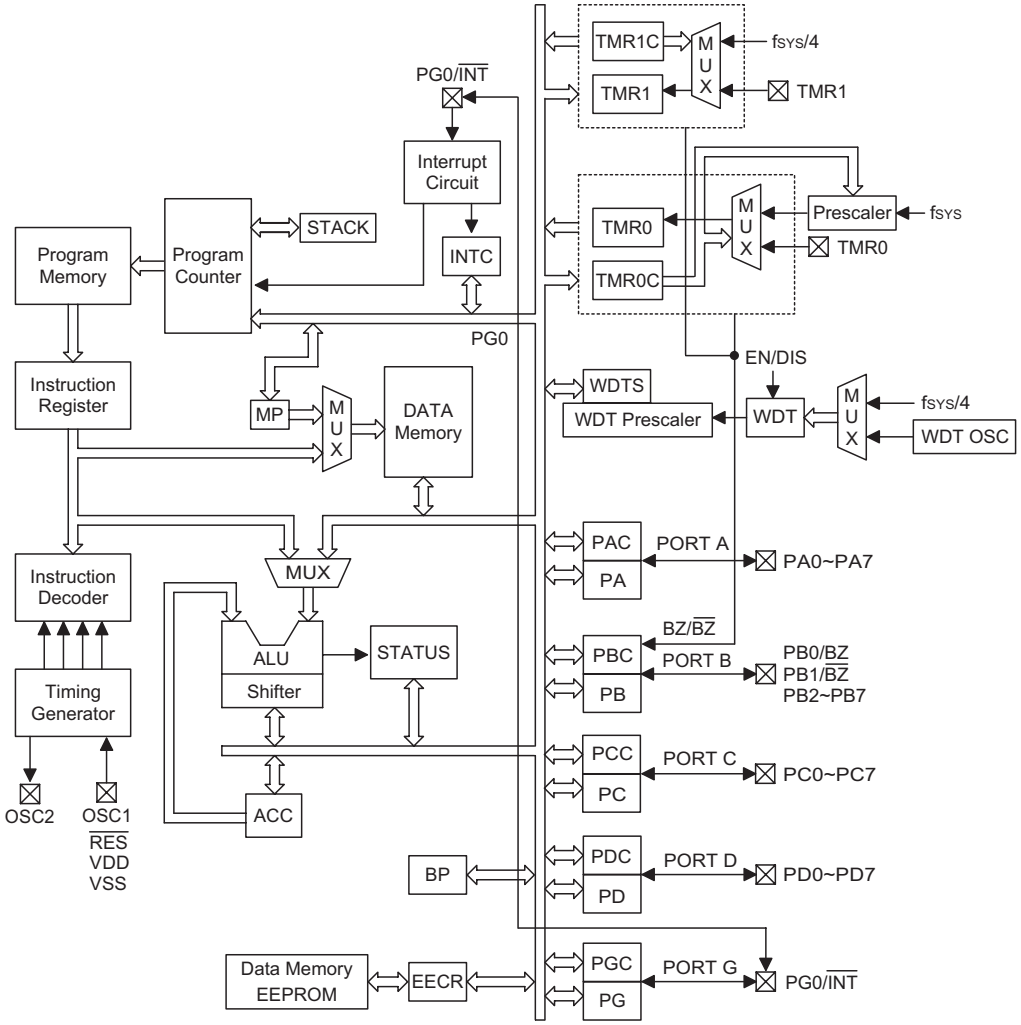
- 工作电压:
  fsys = 4MHz：2.2V – 5.5V
  fsys = 8MHz：3.3V – 5.5V
- 低电压复位功能
- 最多有 33 个双向输入/输出口
- 1 个与输入/输出共用引脚的外部中断输入
- 8 位可编程定时/计数器，具有溢出中断及 8 级预分频器
- 内置晶体和 RC 振荡电路
- 看门狗定时器
- 1,000 次可擦/写 MTP 程序存储器
- 4096×15 程序存储器 ROM（MTP）
- 256×8 数据存储器 EEPROM
- 160×8 数据存储器 RAM
- HALT 和唤醒功能可降低功耗
- 6 层硬件堆栈
- 在 VDD=5V，系统时钟为 8MHz 时，指令周期为 0.5 μs
- 位操作指令
- 查表指令，表格内容字长 15 位
- 63 条指令
- 10^6 次可擦/写 EEPROM 数据存储器
- EEPROM 数据有效期>10 年
- 所有指令在 1 或 2 个指令周期内完成
- 在线烧写功能（ISP）
- 28-pin SKDIP/SOP，48-pinSSOP 的封装

概述

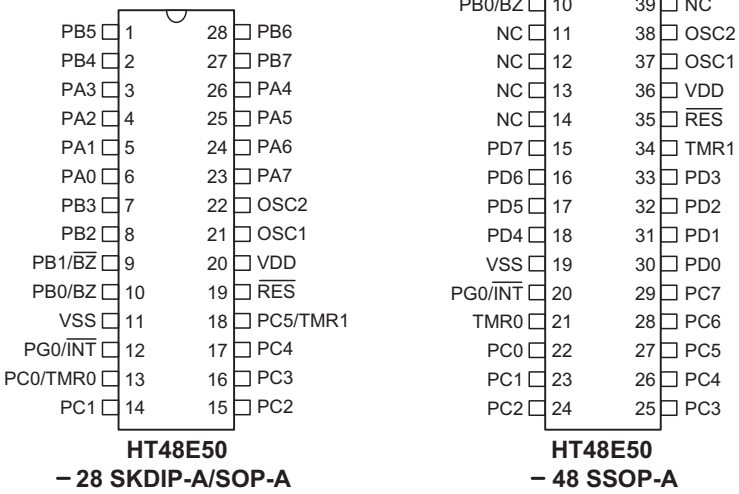
HT48E50 是一款八位高性能精简指令集单片机，专为经济型多输入/输出控制的产品设计。

拥有低功耗、I/O 口稳定性高、定时器功能、振荡选择、省电和唤醒功能、看门狗定时器、蜂鸣器驱动、以及低价位等优势，使此款多功能芯片可以广泛地适用于各种应用，例如工业控制、消费类产品、子系统控制器等。

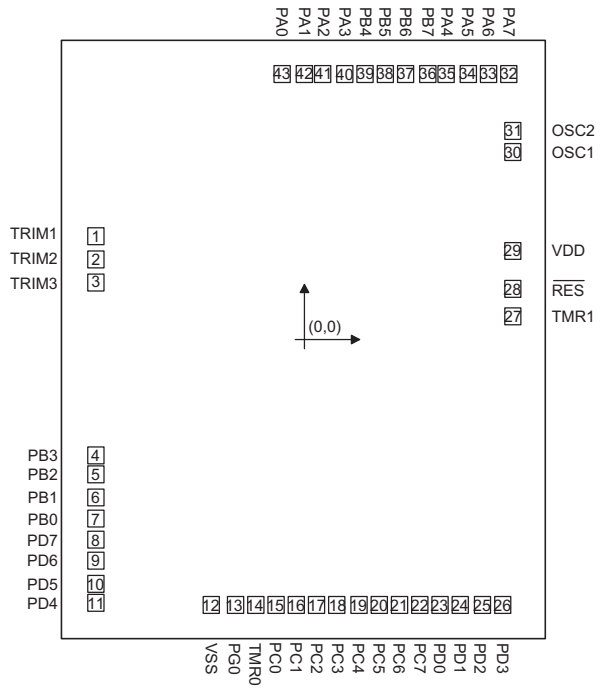
方框图


TMR1C
TMR1
MUX
fSYS/4
TMR1
PG0/INT
Interrupt Circuit
Prescaler
fSYS
TMR0
TMR0C
TMR0
Program Memory
Program Counter
STACK
INTC
PG0
EN/DIS
WDTS
WDT Prescaler
WDT
fSYS/4
WDT OSC
Instruction Register
MP
DATA Memory
Instruction Decoder
MUX
ALU
Shifter
STATUS
Timing Generator
ACC
OSC2
OSC1
RES
VDD
VSS
PAC
PA
PORT A
PA0~PA7
BZ/BZ
PBC
PB
PORT B
PB0/BZ
PB1/BZ
PB2~PB7
PCC
PC
PORT C
PC0~PC7
BP
PDC
PD
PORT D
PD0~PD7
Data Memory EEPROM
EECR
PGC
PG
PORT G
PG0/INT

引脚图



Pad 图



*IC 的衬底要连接到 PCB 板示意图上的 VSS

引脚说明

引脚名称	输入/输出	掩膜选项	说　明
PA0~PA7	输入/输出	上拉电阻* 唤醒功能 CMOS/斯密特触发输入	8 位双向输入/输出口。每一位能由掩膜选项设置为唤醒输入。可由软件指令设置为 CMOS 输出或斯密特触发输入，作为 CMOS 输入时可由掩膜选项设置是否带上拉电阻（由上拉电阻选项决定）。
PB0/BZ PB1/$\overline{BZ}$ PB2~PB7	输入/输出	上拉电阻* PB0 或 BZ PB1 或 $\overline{BZ}$	8 位双向输入/输出口。可由软件指令设置为 CMOS 输出或斯密特触发输入，作为输入时可由掩膜选项设置是否带上拉电阻（由上拉电阻选项决定）。 PB0 和 PB1 是与 BZ 和 $\overline{BZ}$ 共用引脚。一旦 PB0 和 PB1 选为蜂鸣器输出，它的输出信号则由内部的 PFD 发生器（由定时/计数器 0 编程决定）提供。
PD0～PD7	输入/输出	上拉电阻*	双向输入/输出口。可由软件指令设置为 CMOS 输出或斯密特触发输入，作为输入时可由掩膜选项设置是否带上拉电阻（由上拉电阻选项决定）。
V_{SS}	—	—	负电源，接地。
PG0/$\overline{INT}$	输入/输出	上拉电阻*	双向输入/输出口。可由软件指令设置为 CMOS 输出或斯密特触发输入，作为输入时可由掩膜选项设置是否带或上拉电阻（由上拉电阻选项决定）。外部中断输入与 PG0 共用；下降沿触发有效
TMR0	输入	—	定时/计数器 0 斯密特触发输入（不带上拉电阻）。
PC0~PC7	输入/输出	上拉电阻*	双向输入/输出口。可由软件指令设置为 CMOS 输出或斯密特触发输入，作为输入时可由掩膜选项设置是否带或上拉电阻（由上拉电阻选项决定）。
TMR1	输入	—	定时/计数器 1 斯密特触发输入（不带上拉电阻）。
$\overline{RES}$	输入	—	斯密特触发复位输入端。低电平有效。
V_{DD}	—	—	正电源。
OSC1/PG1 OSC2/PG2	输入 输出	上拉电阻* 晶体振荡/ RC 振荡	OSC1 和 OSC2 连接至 RC 或晶体振荡（由掩膜选项确定）来产生内部系统时钟；在 RC 振荡方式下，OSC2 是系统时钟四分频的输出端。

注意：　“*”所有输入/输出(PA、PB、PC、PD、PG)的上拉电阻由一个选择位控制。

PA 端口的 CMOS 或斯密特触发选择也是以整个端口(8 位)为单位的。

极限参数

电源供应电压　…… V_{SS} - 0.3V 至 V_{SS} + 6.0V　　储存温度　…… -50℃至 125℃

端口输入电压　…… V_{SS} - 0.3V 至 V_{DD} + 0.3V　　工作温度　…… -40℃至 85℃

注意：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。

D.C.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	—	fsys=4MHZ	2.2	—	5.5	V
		—	fsys=8MHZ	3.3	—	5.5	V
I_{DD1}	工作电流 (晶体振荡)	3V	无负载 Fsys=4MHZ	—	1	2	mA
		5V		—	3	5	mA
I_{DD2}	工作电流 (RC 振荡)	3V	无负载 fsys=4MHZ	—	1	2	mA
		5V		—	2.5	4	mA
I_{DD3}	工作电流 (晶体振荡，RC 振荡)	5V	无负载 fsys=8MHZ	—	4	8	mA
I_{STB1}	静态电流 (看门狗打开)	3V	无负载 暂停模式	—	—	5	μA
		5V		—	—	10	μA
I_{STB2}	静态电流 (看门狗关闭)	3V	无负载 暂停模式	—	—	1	μA
		5V		—	—	2	μA
V_{IL1}	输入/输出口的低电平输入电压,	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	输入/输出口的高电平输入电压,	—	—	$0.7V_{DD}$	—	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	—	LVR 打开	2.7	3.0	3.3	V
I_{OL}	输入/输出口灌电流	3V	$V_{OL}=0.1V_{DD}$	4	8	—	mA
		5V	$V_{OL}=0.1V_{DD}$	10	20	—	mA
I_{OH}	输入/输出口源电流	3V	$V_{OH}=0.9V_{DD}$	-2	-4	—	mA
		5V	$V_{OH}=0.9V_{DD}$	-5	-10	—	mA
R_{PH}	上拉电阻	3V	—	20	60	100	KΩ
		5V	—	10	30	50	KΩ

A.C.特性

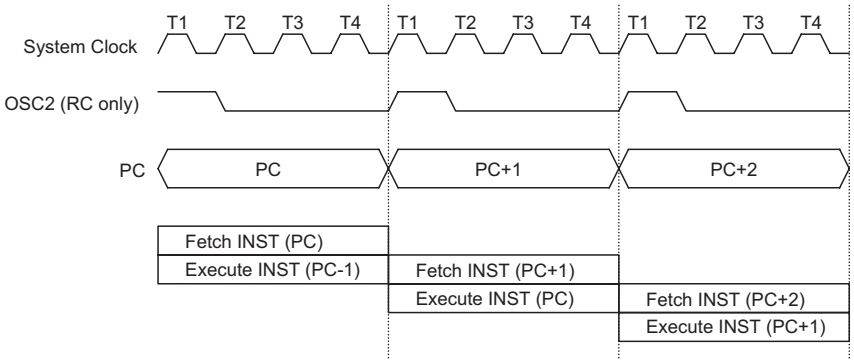
Ta=25℃

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（晶体振荡）	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f_{SYS2}	系统时钟（RC 振荡）	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f_{TIMER}	定时器输入频率 (TMR0/TMR1)	—	2.2V~5.5V	0	—	4000	kHz
		—	3.3V~5.5V	0	—	8000	kHz
t_{WDTOSC}	看门狗振荡器周期	3V	—	45	90	180	μs
		5V	—	32	65	130	μs
t_{WDT1}	看门狗溢出周期(WDT OSC)	3V	WDT 无预分频	11	23	46	ms
		5V		8	17	33	ms
t_{WDT2}	看门狗溢出周期(系统时钟)	—	WDT 无预分频	—	1024	—	t_{SYS}
t_{RES}	外部复位低电平脉宽	—	—	1	—	—	μs
t_{SST}	系统启动延时周期	—	HALT 模式唤醒	—	1024	—	t_{SYS}
t_{INT}	中断脉冲宽度	—	—	1	—	—	μs

系统功能说明

指令系统

系统时钟由晶体振荡器或 RC 振荡器产生。系统内部对此频率进行四分频，产生四个不重叠的时钟周期。一个指令周期包括四个系统时钟周期。



指令执行时序

指令读取与执行是以流水线方式来进行的。这种方式允许在一个指令周期进行读取指令操作，而在下一个指令周期里进行解码与执行该指令。这种流水线方式能在一个指令周期里有效地执行一个指令。但是，如果指令是要改变程序计数器，就需要花两个指令周期来完成这一条指令。

程序计数器（Program Counter — PC）

程序计数器是作为程序存储器寻址之用，控制了程序的流程单片机通过 PC 指向的程序存储器的地址取得一条指令码后， PC 会自动地指向下一条指令的地址（PC 值自动加 1）。

但是若执行的是如下指令：跳转、条件跳跃、读取 PCL 的值、子程序调用或子程序返回、初始复位、内部或外部中断响应或中断返回等，则 PC 要根据每一条指令获得其相应的地址来控制程序的转向。

比如执行条件跳转指令，一旦条件符合，则在当前执行指令时所获取的指令码不会被执行，并且同时会插入一个空的指令周期（dummy cycle），换句话说，相当与执行了一个 NOP 指令（空操作），这样 PC 才会指向正确的指令码的地址；反之，条件不符合时，PC 将指向下一条指令的地址。

PC 的低位（PCL）是可读写的暂存器（06H）。若向 PCL 写入一个值将会产生一个短程的跳跃动作，这种跳转只能在程序存储器的当前页范围内。

当发生控制转移时，就会加入一个空指令周期。

模　　式	程序计数器											
	***11**	***10**	***9**	***8**	***7**	***6**	***5**	***4**	***3**	***2**	***1**	***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	0	0	0	0	0	0	0	0	1	0	0	0
定时/计数器 1 溢出	0	0	0	0	0	0	0	0	1	1	0	0
条件跳转	Program Counter+2											
装载 PCL	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
跳转，子程序调用	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
从子程序返回	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

程序计数器

注意：*11 ~ *0 ： 程序计数器位　　S11~ S0 ： 堆栈寄存器位
　　#11 ~ #0 ： 指令代码位　　@7 ~ @0 ： PCL 位

在线烧写（In System Programing）

在线烧写允许对实际应用电路板上 HT48EXX 单片机进行烧写和重复烧写操作，这样有利于使用者在应用开发实验过程中节省时间和成本。ISP（在线烧写）使用 3 线接口与 HT48EXX 单片机进行连接，对芯片内部的程序存储器和 EEPROM 数据存储器进行重复烧写操作。

引脚名称	功能	说明
PA0	SDATA	串行数据输入/输出
PA4	SCLK	串行时钟输入
RES	RESET	复位
VDD	VDD	电源
VSS	VSS	地

ISP 引脚定义

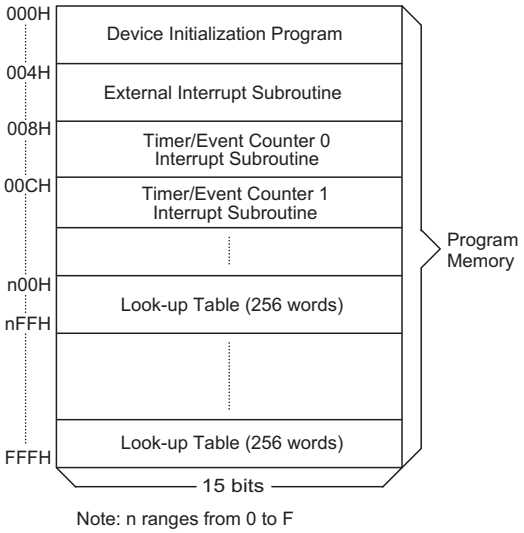
程序存储器（Program Memory － ROM）

程序存储器用来存储要执行的程序指令，也包含数据、表格、中断入口地址，有 4096×15 位，程序存储器空间可以用程序计数器或表格指针进行寻址。

ROM 里面的某些地址是为一些特殊使用而保留的，使用时应加以注意，避免误用，导致程序运行的不正常，以下是说明：

- 地址 000H
  此地址保留给程序初始化之用。当系统复位时，程序会从 000H 地址开始执行。
- 地址 004H
  该地址为外部中断服务程序保留。当$\overline{\text{INT}}$引脚有触发信号输入，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
  此地址保留给定时/计数器 0 中断服务使用。当中断是开放的，且堆栈又未满，则一旦定时/计数器发生溢出时，就能产生中断，程序会从 008H 地址开始执行中断服务程序。
- 地址 00CH
  此地址保留给定时/计数器 1 中断服务使用。当中断允许，且堆栈未满，则一旦定时/计数器发生溢出时，就能产生中断，程序会从 00CH 地址开始执行中断服务程序。
- 表格区（Table Location）

  ROM 内的任何地址都可被用来作为查表地址使用。查表指令为 TABRDC [m] 与 TABRDL [m]。TABRDC [m]是查表当前页的数据 [1 页=256 个字 (word)]。TABRDL [m]是查表最后一页的数据。[m] 为数据被存入的地址。在执行 TABRDC [m]指令（或 TABRDL [m] 指令）后，将会传送当前页（或最后一页）上的一个字的低位字节到[m]，而这个字的高位字节传送到 TBLH（08H）。只有表格中的低位字节被定义到目标地址中，而表格中的高位字节的其它位被传送到 TBLH 的低位部分，剩余的 1 个高位作为 0 读出。TBLH 为只读寄存器。而表格指针（TBLP；07H）是可以读写的寄存器，用来指明表格地址。在访问表格以前，通过对 TBLP 寄存器赋值来指明表格地址。高位字节寄存器 TBLH 只能读出，不能写入。如果主程序和中断服务程序（ISR）同时使用查表指令，那么主程序读取的高位字节（即存放于高位字节寄存器 TBLH 之中）可能会被中断服务程序的查表指令改写而产生错误。因此，应该避免主程序和中断服务程序（ISR）同时使用查表指令。但是，如果主程序和中断服务程序必须同时使用查表指令的话，那么，主程序在使用查表指令之前，必须先关闭所有使用查表指令的相关中断，直到高位字节寄存器 TBLH 的内容被备份好再开放这些中断。查表指令要花两个指令周期来完成这一条指令的操作。按照用户的需要，表格地址这些位置可以作为正常的程序存储器来使用。



Note: n ranges from 0 to F

程序存储器

指令	表格地址											
	***11**	***10**	***9**	***8**	***7**	***6**	***5**	***4**	***3**	***2**	***1**	***0**
TABRDC[m]	P11	P10	P9	P8	@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

注意：*11 ~ *0 ：表格地址位　　P11 ~ P8 ：当前程序计数器位
@7 ~ @0 ：表格指针位

堆栈寄存器（Stack Register — STACK）

堆栈寄存器是用来存放程序计数器 PC 内容的一个特殊的寄存器。HT48E50 有 6 层堆栈，该寄存器既不是数据存储器的一部分又不是程序的一部分，而且也不可读不可写。所进入的级数是由堆栈指针 SP 来指示的，而堆栈指针 SP 也是不可读不可写的。一旦发生了子程序的调用或是中断响应，则程序计数器 PC 的内容会被压入堆栈中。在子程序调用或中断响应结束时（可从返回指令（RET 或 RETI）看出），程序计数器 PC 的值会从堆栈中还原。在系统复位后，堆栈指针 SP 会指向堆栈的顶端。

当堆栈已满，而此时又发生的中断请求，则这个中断的请求标志会被记录下来，该中断服务仍被禁止。一旦堆栈指针 SP 发生了递减（由于 RET 或 RETI）则会响应此未被服务的中断。这个功能就可确保堆栈不会溢出，使得编程者更加方便地使用该结构。同样地，如果堆栈已满，而随后又执行了 CALL 指令，此时会发生堆栈溢出，并且第一个返回地址将会丢失（只有最近的 6 个返回地址会被保存）。

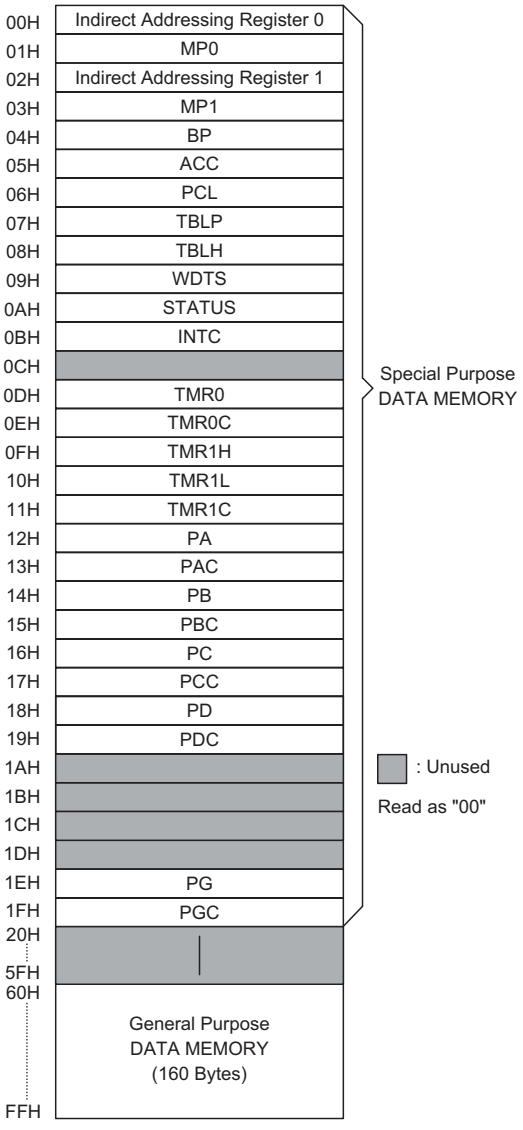
数据存储器（Data Memory — RAM）

数据存储器（RAM）由 184×8 个位组成，包含特别功能寄存器、通用数据寄存器（160×8）两个不同功能区间。这些寄存器大多是可读可写的，只有少部分是只读的。

特殊功能寄存器包括：间接寻址寄存器（R0；00H，R1；02H），存储器段指针（BP；04），定时/计数器0（TMR0；0DH），定时/计数器0控制寄存器（TMR0C；0EH），定时/计数器1高字节（TMR1H；0FH），定时/计数器1低字节（TMR1L；10H），定时/计数器1控制寄存器（TMR1C；11H），程序计数器低字节寄存器（PCL；06H），间接寻址寄存器（MP0；01H，MP1；03H），累加器（ACC；05H），表格指针（TBLP；07H），表格高字节寄存器（TBLH；08H），状态寄存器（STATUS；0AH），中断控制寄存器（INTC；0BH），看门狗选项设置寄存器（WDTS；09H），输入/输出寄存器（PA；12H，PB；14H，PC；16H，PD；18H，PG；1EH），输入/输出控制寄存器（PAC；13H，PBC；15H，PCC；17H，PDC；19H，PGC；1FH）。在60H以前的剩余单元都保留为将来进一步扩展使用。读取这些被保留单元的值，都将返回00H的值。

通用数据存储器的地址从 60H~FFH，作为数据和控制信息使用。

所有的 RAM 都可以直接执行算术、逻辑、递增、递减和移位等运算。除了一些少数指定的位之外，RAM 的每个位都可以由 SET[m].i 和 CLR[m].i 指令来置位和复位。这些 RAM 地址可以通过间接寻址寄存器 MP0（01H）或 MP1（03H）来存取。EEPROM 数据存储器的控制寄存器在 BANK1 的[40H]单元。



数据存储器

间接寻址寄存器（Indirect Addressing Register）

地址 00H 和 02H 作为间接寻址寄存器。它们都没有实际的物理结构。任何对[00H]([02H])的读/写操作，都会访问由 MP0(MP1)所指向的 RAM 单元。间接地读取 00H(02H)单元，将会返回 00H，而间接地写入 00H(02H)单元，则不会产生任何操作。

间接寻址指针(MP0 和 MP1)的宽度都是 8 位，它们与间接寻址寄存器相结合用来间接访问 RAM。MP0 只能应用于数据存储区的 BANK0，而 MP1 能应用于数据存储区的 BANK0 和 BANK1。

累加器（Accumulator）

累加器（ACC）与算术逻辑单元（ALU）有关，同样也是对应至 RAM 的地址 05H，作为运算的立即数据，存储器之间的数据传送必须经过 ACC。

算术逻辑单元（Arithmetic and logic unit — ALU）

算术逻辑单元（ALU）为执行八位算术及逻辑运算的电路，提供有下列的功能：

- 算术运算（ADD，ADC，SUB，SBC，DAA）
- 逻辑运算（AND，OR，XOR，CPL）
- 移位（PL，RR，RLC，RRC）
- 递增及递减（INC，DEC）
- 分支判断（SZ，SNZ，SIZ，SDZ 等）

ALU 不仅可以储存数据运算的结果，还可以改变状态寄存器

状态寄存器（Status Register — STATUS）

状态寄存器（0AH）的宽度为 8 位，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。该寄存器用来记录状态信息，和控制操作流程。

除了 TO 和 PDF 以外，状态寄存器中的位都可用指令来改变，这种情况与其它寄存器一样。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，执行 HALT 指令，或清除看门狗定时器都能改变 TO 和 PDF。

Z，OV，AC 和 C 标志位都反映了最近一次的操作状态。

位	符号	功　　能
0	C	如果在加法运算中结果产生了进位，或在减法运算中结果不发生借位,那么 C 被置位；反之，C 被清除。它也可被一个带进位循环移位指令影响。
1	AC	在加法运算中低四位产生了向高四位进位，或减法运算中低四位不发生从高四位借位，AC 被置位；反之，AC 被清除。
2	Z	算术运算或逻辑运算的结果为零则 Z 被置位；反之，Z 被清除。
3	OV	如果运算结果向最高位进位，但最高位并不产生进位输出，或那么 OV 被置位；反之，OV 被清除。
4	PDF	系统上电或执行了 CLR WDT 指令，PDF 被清除。执行 HALT 指令 PDF 被置位。
5	TO	系统上电或执行了 CLR WDT 指令，或执行 HALT 指令，TO 被清除。WDT 定时溢出，TO 被置位。
6	—	未定义，读出为零
7	—	未定义，读出为零

状态寄存器 Status（0AH）

另外，在进入中断子程序或执行子程序调用时，状态寄存器的内容不会自动压入堆栈。如果状态寄存器的内容是重要的，而且子程序会改变状态寄存器的内容，那么程序员必须事先将其保存好，以免被破坏。

中 断（Interrupt）

本单片机提供一个外部中断和内部定时/计数器中断。中断控制寄存器（INTC；0BH）包含了中断控制位，用来设置中断允许/禁止及中断请求标志。

一旦有中断子程序被服务，所有其它的中断将被禁止（通过清除 EMI 位）。这种机制能防止中断嵌套。这时如有其它中断请求发生，这个中断请求的标志会被记录下来。如果在一个中断服务程序中有另一个中断需要服务的话，程序员可以设置 EMI 位及 INTC 所对应的位来允许中断嵌套服务。如果堆栈已满，该中断请求将不会被响应。即使相关的中断被允许，也要到堆栈指针发生递减时才会响应。如果需要立即得到中断服务，则必须避免让堆栈饱和。

位	符号	功　　能
0	EMI	总中断控制位(1=允许，　0=禁止)
1	EEI	外部中断控制位(1=允许，　0=禁止)
2	ET0I	定时/计数器 0 中断控制位(1=允许，　0=禁止)
3	ET1I	定时/计数器 1 中断控制位(1=允许，　0=禁止)
4	EIF	外部中断请求标志位(1=有，0=无)
5	T0F	定时/计数器 0 中断请求位(1=有，0=无)
6	T1F	定时/计数器 1 中断请求位(1=有，0=无)
7	—	未使用位，读出为零

中断控制寄存器 INTC（0BH）

所有的中断都具有唤醒功能。当一个中断被服务时，会产生一个控制传送：通过将程序计数器（PC）压入堆栈，然后转移到中断服务程序的入口。只有程序计数器的内容能压入堆栈。如果寄存器和状态寄存器的内容会被中断服务程序改变，从而破坏主程序的预定控制，那么程序员必须事先将这些数据保存起来。

外部中断是由 INT 脚上的下降沿触发的，相关的中断请求位（EIF，INTC 的第 4 位）被置位。当中断允许，堆栈也没有满，一个外部中断触发时，那么将会产生地址 04H 的子程序调用。中断请求标志（EIF）和 EMI 位将也会被清除来禁止另外的中断发生。

内部定时/计数器 0 中断是通过置位定时/计数器 0 中断请求标志位（T0F，INTC 的第五位）来初始化的，中断的请求是由定时器溢出产生的。当中断允许，堆栈又未满，并且 T0F 已被置位，就会产生地址 08H 的子程序调用。该中断请求标志位（T0F）被复位并且 EMI 位也将被清除，以便禁止其他中断。

内部定时/计数器 1 中断是通过置位定时/计数器中断请求标志位（T1F，INTC 的第六位）来初始化的，中断的请求是由定时器溢出产生的。当中断允许，堆栈又未满，并且 T1F 已被置位，就会产生地址 0CH 的子程序调用。该中断请求标志位（T1F）被复位并且 EMI 位也将被清除，以便禁止其他中断。

单片机在执行中断子程序期间，其他的中断响应会被暂停，直到执行 RETI 指令或是 EMI 位和相关的中断控制位都被置为 1（当堆栈是未满时）。若要从中断子程序返回时，只要执行 RET 或 RETI 指令即可。RETI 指令将会自动置位 EMI 来再次允许中断服务，而 RET 则不能自动置位 EMI。

如果中断在内部二个连续的 T2 脉冲上升沿间发生，而且中断响应被允许的话，那么在第二个 T2 脉冲，该中断会被服务。如果同时发生中断服务请求，那么下列表中列出了中断服务优先等级。这种优先级也可以通过 EMI 位的复位来屏蔽。

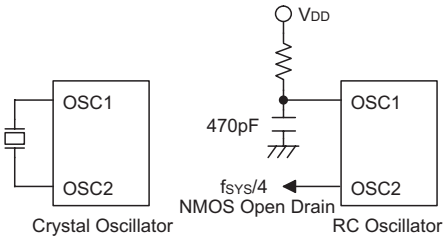
NO	中断源	优先级	中断
a	外部中断	1	04H
b	定时/计数器 0 中断	2	08H
c	定时/计数器 1 中断	3	0CH

中断控制寄存器（INTC）由定时/计数器 0 中断请求标志位（T0F），定时/计数器 1 中断请求标志位（T1F），外部中断请求标志位（EIF），定时/计数器 0 允许位（ET0I），定时/计数器 1 允许位（ET1I），外部中断允许位（EEI），和主中断控制允许位（EMI）组成。EMI、EEI 和 ETI 都是用来控制中断的允许/禁止状态的。这些位防止正在进行的中断服务中的中断请求。一旦中断请求标志位被置位（T0F，T1F，EIF），它们将在 INTC 寄存器中被保留下来，直到相关的中断被服务或由软件指令来清除。

建议不要在中断子程序中使用“CALL”指令来调用子程序，因为中断随时都可能发生，而且需要立刻给予响应。基于上述情况，如果只剩下一个堆栈，若此时中断不能很好地被控制，而且在这个中断服务程中又执行了 CALL 子程序调用，则会造成堆栈溢出，而破坏原先的控制序列。

振荡器（Oscillator Configuration）

HT48E50 有 2 种振荡电路。这 2 种振荡器都是针对系统时钟而设计的，它们是外部 RC 振荡器和外部晶体振荡器，可由掩膜选项设置。不管所选的是哪一种振荡器，其信号都可以支持系统的时钟。进入暂停模式会使系统时钟工作停止。进入暂停模式还可以忽视外部信号的，以降低功耗。



系统振荡器

如果使用 RC 型振荡器，在 OSC1 和 V_{DD} 之间需要一个外部电阻，其阻值范围为 24kΩ 至 1MΩ。在 OSC2 端可获得系统时钟四分频信号，它可用于同步外部逻辑电路。RC 型振荡方式是一种低成本的方式，可是，振荡频率会随着 V_{DD}、温度和制造漂移而不同。因此，在用于需要非常精确振荡频率的计时操作场合，我们并不建议使用 RC 型振荡器。

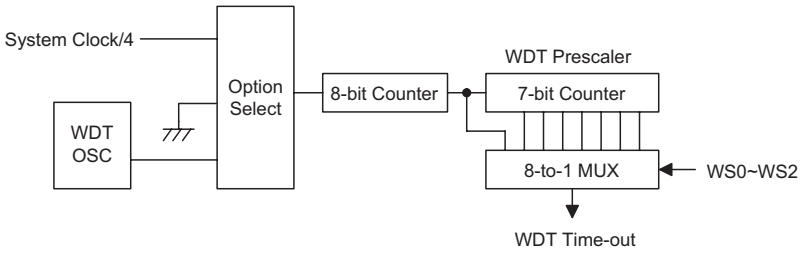
如果选用的是晶体振荡器，那么在 OSC1 和 OSC2 之间需要连接一个晶体，用来提供晶体振荡器所需要的反馈和相移。另外，在 OSC1 和 OSC2 之间还可以用谐振器代替晶体振荡器，来产生系统时钟，但是在 OSC1 和 OSC2 需要多连接两个电容器至地。

WDT 振荡器是 IC 内部自由振荡的 RC 型振荡器，不需要任何外部元件。即使在系统进入暂停模式，系统时钟被停止，但这个 RC 振荡器仍会运作。（其振荡周期大约为 65μs/5V）。如欲节省电源，可在掩膜选项中关闭 WDT 振荡器。

看门狗计时器(Watchdog Timer)

WDT 的时钟源是由一个专用的 RC 振荡器（WDT 振荡器）或是由指令周期（系统时钟 4 分频）来实现的，由掩膜选项来决定的。WDT 是用来防止程序不正常运行或是跳到未知或不希望去的地址，而导致不可预见的结果。WDT 可以被掩膜选项禁止。如果 WDT 定时器被禁止，所有与 WDT 有关的操作都是空操作。

如果设置了内部 WDT 振荡器（以 65μs/5V 为周期的 RC 振荡器）的话，WDT 的值会先除以 256（8 级）来产生大约 17ms/5V 的溢出时间，这个溢出时间会因为温度，V_{DD}，以及芯片参数的变化而变化。如果启动 WDT 的预分频器，则可实现更长的溢出时间。写数据到 WS2,WS1,WS0 (WDTS 的 2、1、0 位)会产生不同的溢出时间，举例来说，如果 WS2，WS1，WS0 的值都为 1，其分频级数最大，为 1：128，溢出时间最长约 2.1s/5V。如果 WDT 被禁止，那么 WDT 的时钟来源可来自指令时钟，其运作与 WDT 振荡器一样。但当在 HALT 状态时，来源于指令时钟的 WDT 会在暂停模式时停止计数并失去保护功能。在这种情况下，只能由外部逻辑来重新启动系统。WDTS 的高四位及其第 3 位保留给用户定义标志来使用，程序员可以利用这些标志来指示某些特殊的状态。



看门狗定时器

如果单片机工作在干扰很大的环境中，那么强烈建议使用片内 RC 振荡器（WDT OSC），因为 HALT 模式会使系统时钟终止运作。

在正常运作下，WDT 溢出会使系统复位并设置 TO 状态位。但在 HALT 模式下，溢出只产生一个“热复位”，只能使 PC 程序计数器和堆栈指针 SP 复位到零。要清除 WDT 的值（包括 WDT 预分频器）可以有三种方法：外部复位（低电平输入到 RES 端），用软件指令和 HALT 指令三种。软件指令由 CLR WDT 和另一组指令 CLR WDT1 及 CLR WDT2 组成。这两组指令中，只能选取其中一种。选择的方式由掩膜选项的 CLR WDT 次数选项决定。如果“CLR WDT”被选择（即 CLR WDT 次数为 1），那么只要执行 CLR WDT 指令就会清除 WDT。在 CLR WDT1 和 CLR WDT2 被选择的情况下（即 CLR WDT 次数为 2），那么要执行二条指令才会清除 WDT。否则，WDT 会由于溢出而使系统复位。

WS2	WS1	WS0	分频
0	0	0	1：1
0	0	1	1：2
0	1	0	1：4
0	1	1	1：8
1	0	0	1：16
1	0	1	1：32
1	1	0	1：64
1	1	1	1：128

看门狗定时器预置寄存器 WDTS（09H）

暂停模式（HALT）

暂停模式是由 HALT 指令来实现的，产生如下结果：

- 关闭系统振荡器，但 WDT 振荡器继续工作（如果 WDT 时钟来源是 WDT 振荡器）。
- RAM 及寄存器的内容保持不变。
- WDT 被被清除并再次重新计数（如果 WDT 时钟来源是 WDT 振荡器）。
- 所有的输入/输出口都保持其原先状态。
- PDF 标志位被置位，TO 标志位被清零。

由于外部复位、中断、外部输入一个下降沿的信号到 PA 口或 WDT 溢出，可使系统脱离暂停状态。外部复位能使系统初始化而 WDT 溢出使系统“热复位”。测试 TO 和 PDF 状态后，系统复位的原因就可以被确定。PDF 标志位是由系统上电复位和执行 CLR WDT 指令被清除，而它的置位是由于执行了 HALT 指令。如果 WDT 产生溢出，会使 TO 标志位置位，还能产生唤醒使得程序计数的 PC 和堆栈指针 SP 复位。其他都保持原状态。

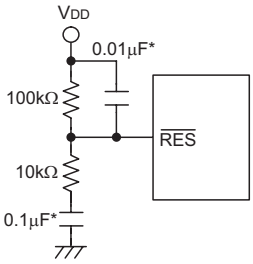
PA 口的唤醒和中断方式被看作为正常运行。PA 口的每一位都可以通过掩膜选项来单独设定为对系统的唤醒。如果唤醒是来自于输入/输出口的信号变化，程序会重新继续执行下一条命令。如果唤醒是来自于中断，那么有两种情况可能发生：如果相关的中断被禁止或中断是允许的，但堆栈已满，那么程序将继续执行下条指令，如果中断允许并且堆栈未满，那么这个中断响应就发生了。如果在进入 HALT 模式以前，中断请求标志位被置“1”，那么相关的中断唤醒功能被禁止。一旦唤醒事件发生，要花 1024tsys（系统时钟周期），系统重新正常运行。换句话说，在唤醒后被插入了一个等待时间。如果唤醒是来自于中断响应，那么实际的中断程序执行就被延迟了一个或一个以上的周期。但是如果唤醒导致下一条指令执行，那么在一个等待周期结束后指令就立即被执行。

为了省电，在进入 HALT 模式之前必须要小心处理输入/输出口状态。

复位（Reset）

有三种方法可以产生复位：

- 在正常运行时期由$\overline{\text{RES}}$脚产生复位。
- 在 HALT 期间$\overline{\text{RES}}$脚产生复位。
- 正常运行时，WDT 溢出复位。



复位电路

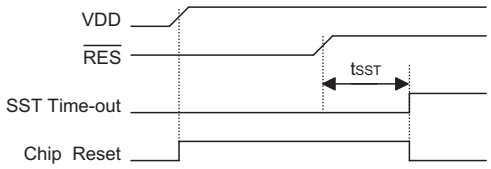
注意：“*”为了避免噪声干扰，连接$\overline{\text{RES}}$引脚的线应尽量短

在 HALT 期间 WDT 溢出是不同于其它的复位操作条件，因为它可执行 “热复位”，结果只能使程序计数器 PC 和堆栈指针 SP 复位，而别的寄存器均保持原来的状态。在其他复位条件下，某些寄存器保持不变。当复位条件被满足时，极大多数的寄存器被复位到“初始状态”，通过测试 PDF 标志位和 TO 标志位，程序能分辨不同的系统复位。

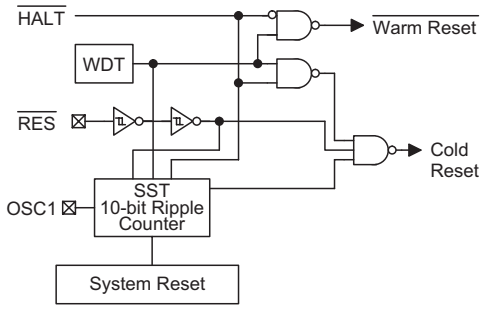
TO	PDF	复 位 条 件
0	0	电源上电复位
u	u	正常运作时由$\overline{\text{RES}}$复位
0	1	$\overline{\text{RES}}$时唤醒暂停模式
1	u	正常运作时发生看门狗定时器超时
1	1	由看门狗定时器唤醒暂停模式

注释：“u”表示未变化。

为保证系统振荡器起振并稳定运行，那么 SST（系统启动定时器）当系统复位（上电，WDT 定时溢出或$\overline{\text{RES}}$）或从 HALT 状态被唤醒时，提供额外延迟 1024 个系统时钟脉冲。



复位时序



复位电路结构

当系统复位时，SST 延迟被加到复位周期中。任何来自 HALT 的唤醒都将允许 SST 延迟。

系统复位时各功能单元的状态如下所示：

程序计数器（PC）	000H
中断	禁止
预分频器	清除
看门狗定时器	清除，复位后看门狗定时器开始计数
定时/计数器 0/1	关闭
输入/输出口	输入模式
堆栈指针 SP	指向堆栈顶端

有关寄存器的状态如下：

寄存器	上电复位	正常运行期间		暂停模式	
		WDT 溢出	RES 端复位	RES 端复位	WDT 溢出*
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR1H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
PC	000H	000H	000H	000H	000H
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--lu uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
WDTS	0000 0111	0000 0111	0000 0111	0000 0111	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PG	---- ---1	---- ---1	---- ---1	---- ---1	---- ---u
PGC	---- ---1	---- ---1	---- ---1	---- ---1	---- ---u
EECR	1000 ----	1000 ----	1000 ----	1000 ----	uuuu ----

注意： “*”表示“热复位。

“U”表示不变化。

“×”表示不确定。

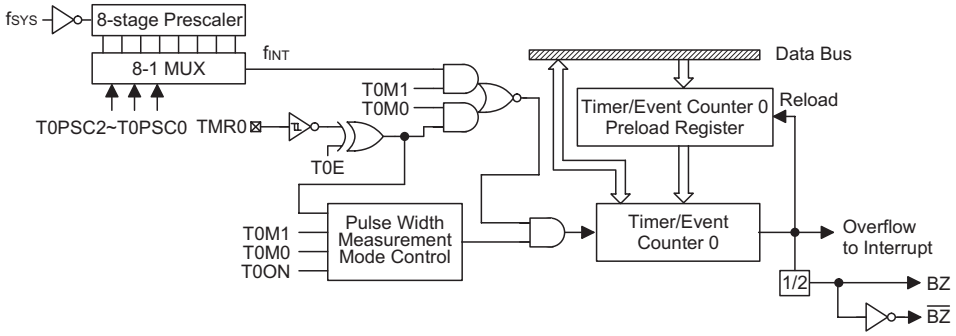
定时/计数器（Timer/Event Counter）

HT48E50 提供两个定时/计数器。定时/计数器 0 包含一个 8 位可编程的向上计数的计数器，并且其时钟来源可以是外部信号输入或是系统时钟。

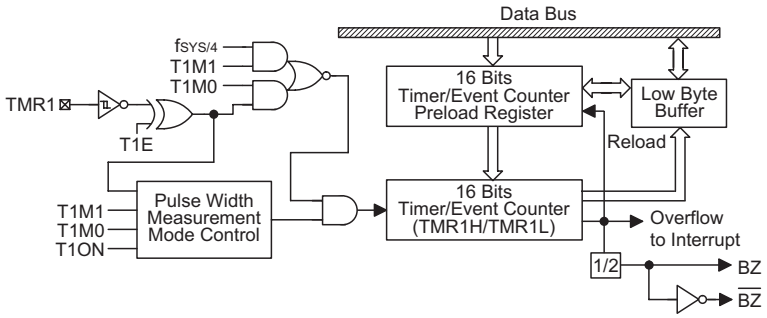
定时/计数器 1 则包含一个 16 位的可编程的向上计数的计数器，且其时钟来源可来至外部信号输入或是指令时钟。

外部时钟输入，允许用户去计数外部事件，测量时间长度或脉冲宽度或产生一个精确的时基信号。

定时/计数器 0 可以产生 PFD 信号，时钟源来自外部或者内部时钟，PFD 频率= $f_{INT}/[2\times(256-N)]$。



定时/计数器 0



定时/计数器 1

有两个寄存器与定时/计数器 0 相关联，即 TMR0（[0DH]）和 TMR0C（[0EH]）。有两个物理寄存器对应 TMR0 的位置。写入 TMR0 会将初始值装入到定时/计数器 0 的预置寄存器中，而读 TMR0 则会获得定时/计数器 0 的内容。TMR0C 是定时/计数器控制寄存器。

TMR0C 是定时/计数器 0 的控制寄存器，用于定义操作模式，计数打开或关闭和触发边沿。

有三个寄存器与定时/计数器 1 相关联，即 TMR1H（[0FH]）和 TMR1L（[10H]）；TMR1C（[11H]）。有两个物理寄存器对应 TMR 的位置。若写入 TMR1L 只能将数据写入低字节内部缓冲器（8 bit）中，但若写入的是 TMR1H 则可将数据和低字节内部缓冲器的内容写入 TMR1H 和 TMR1L 的加载寄存器之中。每一次对 TMR1H 的写操作都会改变定时/计数器加载寄存器的内容。若读取 TMR1H 则将锁存 TMR1H 的内容并将 TMR1L 传送至低字节内部缓冲器之中，以避免发生计时错误。然而，若读取 TMR1L，则只读回低字节内部缓冲器的内容。换言之，定时/计数器的低字节数据并不能直接读取。若欲读取该低字节的数据，必须先读取 TMR1H，以便将定时/计数器的低字节数据传送至内部低字节缓冲器之中。TMR1C 是定时/计数器 1 控制寄存器，它可定义：工作模式、计数功能打开或关闭、计数的触发沿。

T0M0/T1M0(TMR0C)和 T0M1/T1M1(TMR1C)位定义工作模式。外部事件计数模式用来记录外部事件，它的时钟来自外部（TMR0 / TMR1）引脚。定时器模式是一个常用功能，时钟源来自 f_{INT} 时钟或指令时钟（TIMER0/TIMER1）。脉冲宽度测量模式能用来测量外部引脚（TMR0/TMR1）上的高电平或低电平的宽度。计数是基于 f_{INT} 时钟或指令时钟（TIMER0/TIMER1）。

在外部事件计数或定时器模式中，一旦定时/计数器 0/1 开始计数，它将会从当前定时/计数器 0/1 中的数值开始向上计数到 0FFH 或是 0FFFFH。一旦产生溢出，计数器会从定时/计数器 0/1 预置寄存器重新装载，并置位中断请求状态位（T0F/T1F ； INTC 的第 5/6 位）。

在脉冲宽度测量中，将 T0ON/T1ON 和 T0E/T1E 置为“1”，如果 TMR0/TMR1 接收到上升沿（或下降沿，如果 T0E/T1E 位被清零），就开始计数，直到 TMR 返回到原来的电平，并复位 T0ON/T1ON 位。测量

的结果被保留在定时/计数器 0/1 中，即使电平跳变再一次发生也不会改变。换句话说，一次只能测量一个脉宽。直到 T0ON/T1ON 再次被置位，这样再次收到跳变信号，测量会再次执行。要注意在这个操作模式中，定时/计数器 0/1 的启动计数不是根据逻辑电平，而是依据信号的边沿跳变触发。一旦发生计数器溢出，计数器会从定时/计数器 0/1 的预置寄存器重新装入，并引发出中断请求，这种情况与其另外两个模式一样。要使得计数运行，只要将定时器启动位（T0ON；TMR0C 的第 4 位；T1ON；TMR1C 的第 4 位）置 1。在脉宽测量模式中，T0ON/T1ON 在测量周期结束后自动被清零。但在另外两个模式中，T0ON/T1ON 只能由指令来复位。定时/计数器 0/1 的溢出是唤醒的信号之一。不管任何模式，若写 0 到 ET0I/ET1I 位即可禁止相应的中断响应。

在定时/计数器 0/1 为关闭的状态下，写数据到定时/计数器 0/1 的预置寄存器之中，同时也会将数据装入定时/计数器 0/1 中。但若是定时/计数器 0/1 已经开始运行，写到定时/计数器 0/1 的数据只会被保留在定时/计数器 0/1 的预置寄存器中，直到定时/计数器 0/1 发生计数溢出为止，再由预置寄存器加载新的值。读取定时/计数器 0/1（TMR0/TMR1）时，计数会被停止，以避免发生错误；计数停止会导致计数错误，程序员必须注意到这一点。

TMR0C 的 0~2 位被用于定义定时/计数器的内部时钟源的预分频级数。定义如表所示。定时/计数器 0 的溢出信号被用于产生驱动蜂鸣器的 PFD 信号。

位	符号	功能
0-2	T0PSC0~ T0PSC2	定义预分频器级数，T0PSC2，T0PSC1，T0PSC0= 000：f_{INT}= fsys/2 001：f_{INT}= fsys/4 010：f_{INT}= fsys/8 011：f_{INT}= fsys/16 100：f_{INT}= fsys/32 101：f_{INT}= fsys/64 110：f_{INT}= fsys/128 111：f_{INT}= fsys/256
3	T0E	定义定时/计数器 TMR0 的触发方式: （0 = 上升沿作用， 1 = 下降沿作用）
4	T0ON	打开/关闭定时/计数器(1=打开，0=关闭)
5	—	未用，读出为 0
6 7	T0M0 T0M1	定义操作模式 01=外部事件计数模式(外部时钟) 10=定时模式(内部时钟) 11=脉冲宽度测量模式 00 =未用

TMR0C（0EH）寄存器

位	符号	功能
0 － 2	—	未定义，读出为“0”
3	T1E	定义定时/计数器 TMR1 的触发方式 （0=上升沿作用，1=下降沿作用）
4	T1ON	打开/关闭定时/计数器(1=打开，0=关闭)
5	—	未用，读出为 0
6 7	T1M0 T1M1	定义操作模式 01=事件计数模式(外部时钟) 10=定时模式(内部时钟) 11=脉冲宽度测量模式 00 =未用

TMR1C（11H）寄存器

输入/输出口（Input/Output Ports）

单片机具有 33 个双向输入输出口，标号从 PA 到 PD 以及 PG，其分别对应的 RAM 的[12H]，[14H]，[16H]，[18H]和[1EH]。所有的输入/输出端口都能被作为输入或输出使用。输入时，端口不具有锁存功能，即输入数据必须在“MOV A，[m]”(m=12H，14H，16H，18H 或 1EH) 指令的 T2 上升沿被准备好。输出时，，所有的数据被锁存并保持不变，直到执行下一个写入操作。

每个 I/0 口都有其自己的控制寄存器（PAC，PBC，PCC，PDC，PGC），用来控制输入/输出的设置。使用控制寄存器，可对 CMOS 输出或带或不带上拉电阻结构的斯密特触发输入可在软件下动态地进行改变。要设置为输入功能，相应的控制寄存器必须写“1”。信号源的输入也取决于控制寄存器。如果控制寄存器的某位值为“1”那么输入信号是读取自这个引脚（PAD）的状态，但是如果控制寄存器的某位值为“0”，那么锁存器的内容将会被送到内部总线。后者，可以在“读改写”指令中发生。

对于输出功能，只能设置为 CMOS 输出。这些控制寄存器是对应于内存的 13H，15H，17H，19H 和 1FH 地址。

系统复位后，这些输入/输出口都会是高电平或浮空状态（取决于上拉电阻的选项）。每一个输入/输出锁存位都能被 SET [m].i 或 CLR [m].i 指令置位或清零，（m=12H，14H，16H，18H 或 1EH。）

某些指令会首先输入数据然后进行输出操作。例如，SET [m].i ，CLR [m].i，CPL [m]和 CPLA [m]指令，读取输入口的状态到 CPU，执行这个操作（位操作），然后将数据写回锁存器或累加器。

PA 的每一个口都具有唤醒系统的能力。PG 端口的高 7 位在物理上是不存在的；读这些位将返回“0”，而写这些位是空操作。请看应用注释。

所有的输入/输出口都可以有上拉电阻的选择。一旦选择上拉电阻，所有的输入/输出口都具有上拉电阻。必须要注意的是：没有上拉电阻的输入/输出口工作在输入模式会产生浮空状态。

PB0 和 PB1 分别与 BZ 和$\overline{BZ}$管脚复用。如果 BZ/$\overline{BZ}$的选项被选择，PB0/PB1 在输出模式时的输出信号将是由定时/计数器的溢出信号产生的 PFD 信号。在输入模式始终保持它的原来的功能。一旦 BZ/$\overline{BZ}$的选项被选择，蜂鸣器的输出信号只受 PB0 数据寄存器控制。

PB0/PB1 的输入/输出功能如下所示：

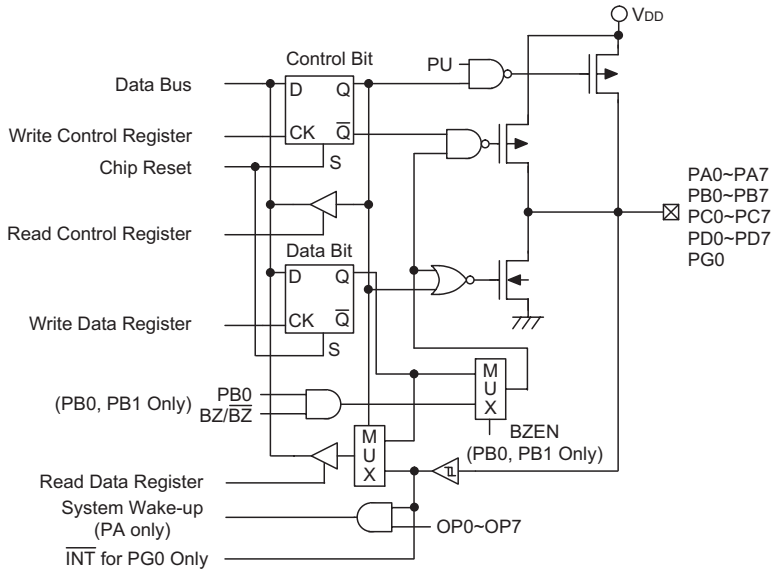
PB0 输入/输出	I	I	O	O	O	O	O	O	O	O
PB1 输入/输出	I	O	I	I	I	O	O	O	O	O
PB0 模式	×	×	C	B	B	C	B	B	B	B
PB1 模式	×	C	×	×	×	C	C	C	B	B
PB0 数据	×	×	D	0	1	D_0	0	1	0	1
PB1 数据	×	D	×	×	×	D_1	D	D	×	×
PB0 Pad 状态	I	I	D	0	B	D_0	0	B	0	B
PB1 Pad 状态	I	D	I	I	I	D_1	D	D	0	B

注意： I：输入； O：输出； D，D0，D1：数据；

B：蜂鸣器的选项，BZ 或$\overline{BZ}$； ×：任意值；

C：CMOS 输出；

PG0 与$\overline{\text{INT}}$管脚复用。



输入/输出口

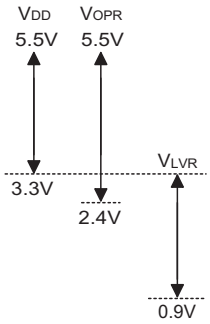
为了避免在浮空状态下功耗太大，建议将未用的或没有连结到外部的输入/输出口由软件指令设置成输出引脚。

低电压复位 （Low Voltage Reset — LVR）

为了监控器件的工作电压，单片机提供低电压复位电路。如果器件的工作电压在 0.9V～V_{LVR} 之间，例如电池电压的变化，那么 LVR 会自动使器件产生内部复位

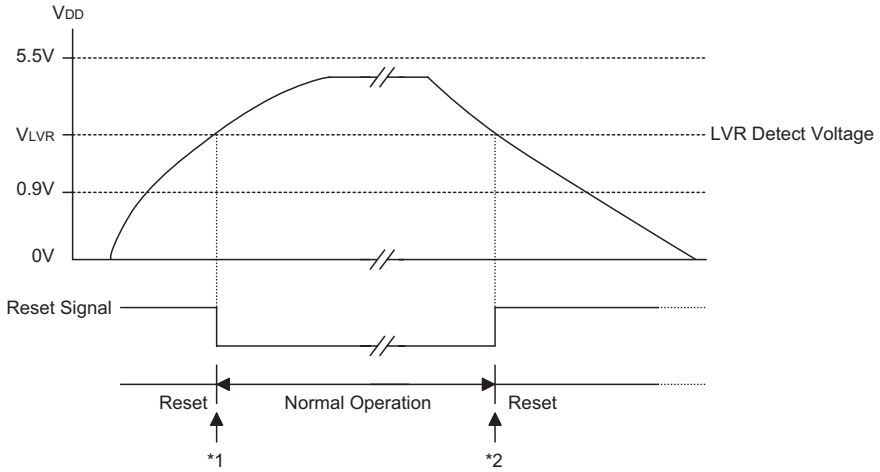
LVR 具有下列功能说明：

- 低电压（0.9 伏～V_{LVR} 伏）的状态必须持续 1ms 以上。如果低电压的状态没持续 1ms 以上，那么 LVR 会忽视它而不去执行复位功能。
- LVR 通过与$\overline{RES}$信号的“或”的功能来执行系统复位。



注：V_{OPR} 为系统时钟 4MHz 时一般芯片正常工作电压的范围。

V_{DD} 与 V_{LVR} 之间的关系如下所示：



低电压复位

注意：

*1： 要保证系统振荡器起振并稳定运行，在系统进入正常运行以前，SST 提供额外的 1024 个系统时钟周期的延迟。

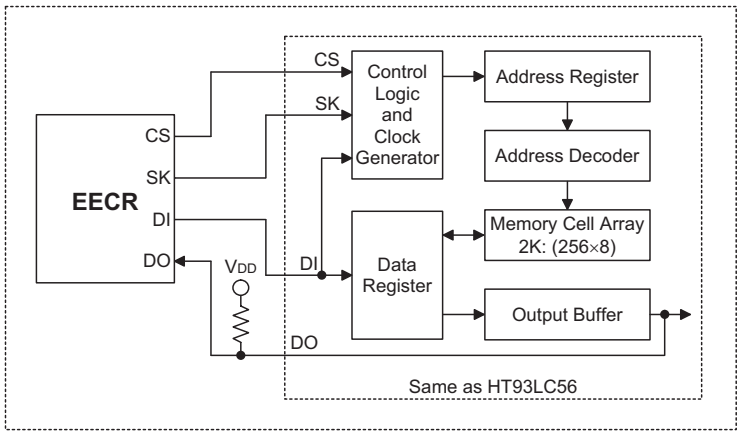
*2： 因为低电压状态必须保持 1ms 以上，因此进入复位模式就要有 1ms 的延迟。

EEPROM 数据存储器（EEPROM Data Memory）

EEPROM 数据存储器包含 256×8 位，在正常工作中可读可写。它要通过控制寄存器 EECR（[40H]在 Bank 1）间接访问。寄存器 EECR 只能使用 MP1 间接读写。

位	标识	功能
0-3	——	未定义，读出为“0”
4	CS	EEPROM 数据存储器片选
5	SK	EEPROM 数据存储器串口时钟输入
6	DI	EEPROM 数据存储器串口数据输入
7	DO	EEPROM 数据存储器串口数据输出

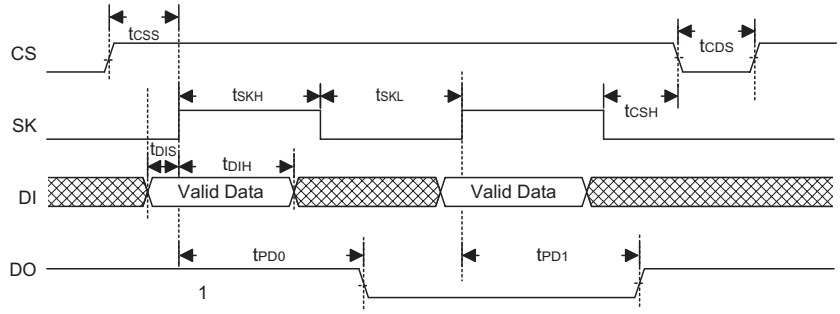
EECR（40H）寄存器



EEPROM 数据存储器方框图

访问 EEPROM 数据存储器需要三根串行通讯线，通过 EECR 界面写入。EEPROM 数据存储器有 256 个字，每个字包含 8 个位。EEPROM 数据存储器有 7 个指令：READ，ERASE，WRITE，EWEN，EWDS，ERAL 和 WRAL。这些指令包含 12 个位：1 个起始位，2 个 op-code 位和 9 个地址位。

通过写 CS,SK 和 DI，这些指令可以被输入到 EEPROM 中。这些在 DI 线上的串行指令数据会在 SK 上升沿写入 EEPROM 数据存储器。在读取周期中，DO 作为数据输出线，在 WRITE 或者 ERASE 周期，DO 处于忙/准备状态。当 DO 作为输入数据线或者处于忙/准备状态，CS 管脚必须置高；反之 DO 会处于一个高状态。指令发送成功之后，CS 必须马上置为低电平。上电后，系统默认处于 EWDS 状态。执行 ERASE 或者 WRITER 指令之前，需要先执行 EWEN 指令。



EECR A.C.特性 Ta=25℃

符号	参数	Vcc=5V±10%		Vcc=2.2V±10%		单位
		最小	最大	最小	最大	
f_{SK}	时钟频率	0	2	0	1	MHz
t_{SKH}	SK 高电平时间	250	—	500	—	ns
t_{SKL}	SK 低电平时间	250	—	500	—	ns
t_{CSS}	CS 建立时间	50	—	100	—	ns
t_{CSH}	CS 保持时间	0	—	0	—	ns
t_{CDS}	CS 屏蔽时间	250	—	250	—	ns
t_{DIS}	DI 建立时间	100	—	200	—	ns
t_{DIH}	DI 保持时间	100	—	200	—	ns
t_{PD1}	DO 等待“1”时间	—	250	—	500	ns
t_{PD0}	DO 等待“0”时间	—	250	—	500	ns
t_{SV}	状态有效时间	—	250	—	250	ns
t_{HZ}	DO 释放时间	100	—	200	—	ns
t_{PR}	每个 WORD 写周期	—	2	—	5	ms

接下来详细描述所有 7 条指令的功能。

READ

读取指令会将指定地址的数据送出到 DO 线上。在 SK 上升沿时，DO 线上的数据会跳变。8 位数据之前有一个逻辑“0”信号。无论 EWEN 或者 EWDS 指令执行与否，读取指令都可以有效执行。一个数据字被读取之后，EEPROM 内地址会自动加一，允许下一个连续的数据被读取，而不需要输入任何地址数据。地址数据会循环累加直到 CS 从高电平变为低电平。

EWEN/EWDS

EWEN/EWDS 指令可以使能或者禁止擦写动作有效执行。无论是处于上电或者掉电状态，芯片都会自动进入禁止擦写模式。执行 WRITE,ERASE,WRAL 或者 ERAL 指令之前，必须先执行 EWEN 指令使能擦写动作有效执行，反之，执行 ERASE/WRITE 指令就是无效的。EWEN 指令运行后，擦写动作使能状态就被启动，直到断电或者 EWDS 指令被运行。当处于禁止擦写状态时，就无法向 EEPROM 数据存储器进行擦写。这样做的话，内部数据就相当于被保护了。

ERASE

在擦写使能状态下，ERASE 指令可以擦除指定地址的数据内容。ERASE 指令码和指定地址送出后，数据在 CS 下降沿会被清除。由于芯片内部有自动时序发生器，提供所有时序信号给内部擦除动作，所以 SK 时钟不需要外部提供。内部擦除动作期间，当 CS 保持高电平我们可以检测到系统状态是忙还是空闲。忙状态时 DO 线保持低电平，直到擦除动作结束，DO 会恢复到高电平，此时其它指令可以被执行。

WRITE

擦写使能模式下，WRITE 指令可以将数据写入指定地址的 EEPROM 数据存储器。WRITE 指令码和指定地址送出后，数据在 CS 下降沿会被写入。由于芯片内部有自动时序发生器，提供所有时序信号给内部写入动作，所以 SK 时钟不需要外部提供。整个自动写入周期包括，擦除然后再写入。因此不需要在写入指令前擦除数据。内部写入动作期间，当 CS 保持高电平我们可以检测到系统状态是忙还是空闲。忙状态时 DO 线保持低电平，直到擦除动作结束，DO 会恢复到高电平，此时其它指令可以被执行。

ERAL

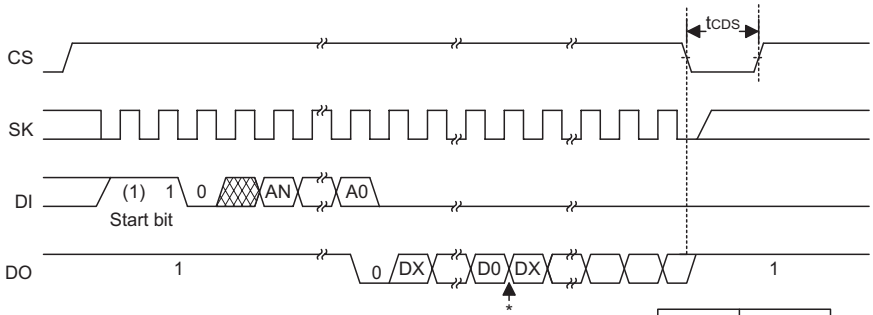
在擦写使能状态下，ERAL 指令可以擦除所有 256×8 存储器位单元为逻辑“1”。擦除所有内容指令送出后，数据在 CS 下降沿会被清除。由于芯片内部有自动时序发生器，提供所有时序信号给擦除所有内容动作，所以 SK 时钟不需要外部提供。内部擦除所有内容动作期间，当 CS 保持高电平我们可以检测到系统状态是忙还是空闲。忙状态时 DO 线保持低电平，直到擦除动作结束，DO 会恢复到高电平，此时其它指令可以被执行。

WRAL

在擦写使能状态下，WRAL 指令可以写入所有 256×8 存储器。写入所有内容指令送出后，数据在 CS 下降沿会被写入。由于芯片内部有自动时序发生器，提供所有时序信号给写入所有内容动作，所以 SK 时钟不需要外部提供。内部写入所有内容动作期间，当 CS 保持高电平我们可以检测到系统状态是忙还是空闲。忙状态时 DO 线保持低电平，直到擦除动作结束，DO 会恢复到高电平，此时其它指令可以被执行。

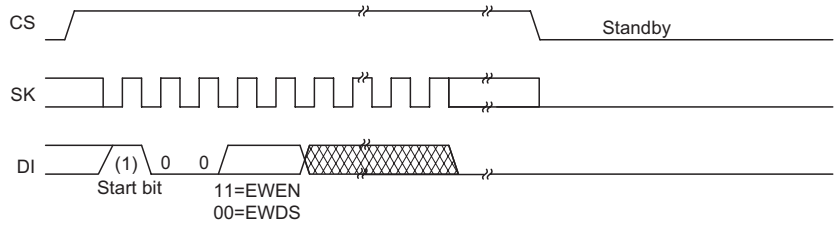
EECR 控制时序图

- READ

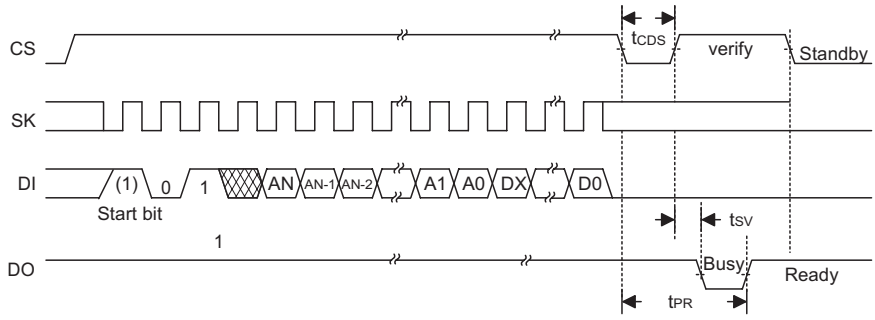


Mode	(X8)
AN	A7
DX	D7

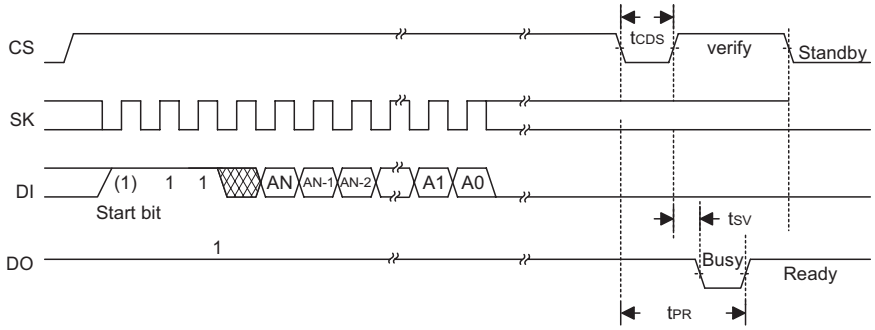
- EWEN/EWDS



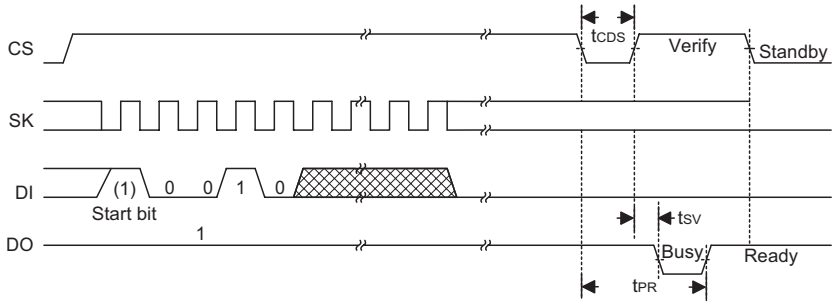
- WRITE



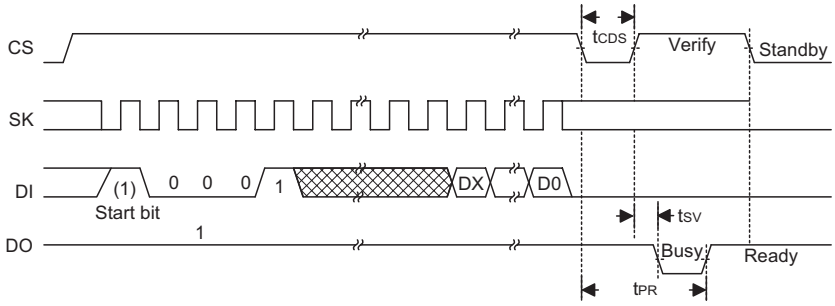
- ERASE



- ERAL



- WRAL



EEPROM 数据存储器指令设置概要

指令	描述	起始位	指令码	地址	数据
READ	Read Data	1	10	X,A7~A0	D7~D0
ERASE	Erase Data	1	11	X,A7~A0	——
WRITE	Write Data	1	01	X,A7~A0	D7~D0
EWEN	Erase/Write Enable	1	00	11XXXXXXX	——
EWDS	Erase/Write Disable	1	00	00XXXXXXX	——
ERAL	Erase All	1	00	10XXXXXXX	——
WRAL	Write All	1	00	01XXXXXXX	D7~D0

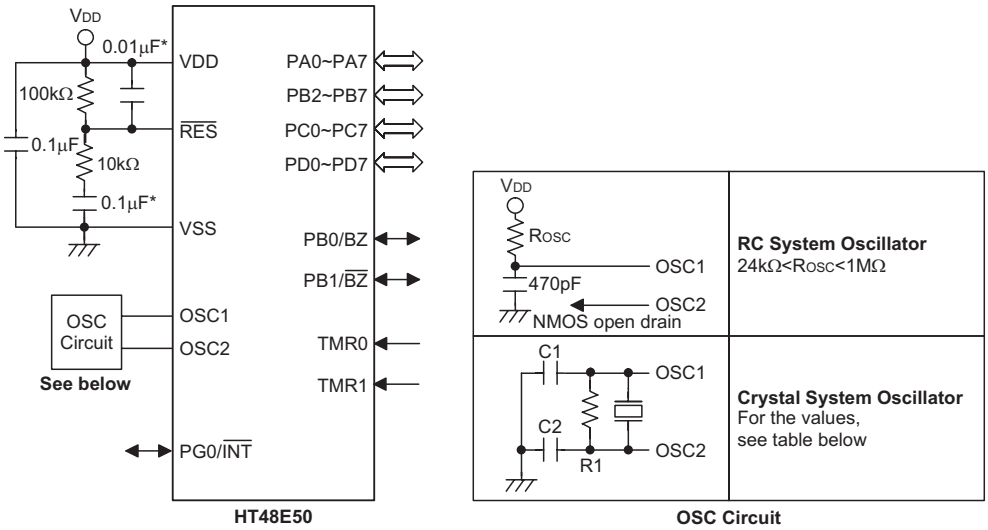
注意：”X”表示不确定

掩膜选项（Options）

下表列出所有掩膜选项。在系统运行前必须定义所有选项的值。

编号	选　　项
1	WDT 时钟源：WDT 振荡或 Fsys / 4 或关闭
2	清除看门狗指令条数：1 或 2 条指令
3	定时/计数器 0 时钟来源：f_{SYS}
4	定时/计数器 1 时钟来源：$f_{SYS}/4$
5	PA 唤醒功能：有/没有（位选择）
6	PA 端口：CMOS/斯密特输入
7	上拉电阻（PA、PB、PC、PD、PG）：有/没有（端口选择）
8	BZ 选项：打开/关闭
9	LVR：打开/关闭
10	系统振荡器：外部 RC 振荡/外部晶体振荡
11	BZ/$\overline{BZ}$ 时钟源：TMR0/TMR1

应用电路



注意： 电阻和电容值选取的原则是使 VDD 保持稳定并在$\overline{\text{RES}}$置为高以前把工作电压保持在允许的范围内。

“*”为了避免噪声干扰，连接$\overline{\text{RES}}$引脚的线请尽可能地短。

下表所示为根据不同的晶振值选择 R1、C1、C2

晶体振荡或谐振器	C1，C2	R1
4MHz 晶振	0pF	10KΩ
4MHz 谐振器（3 脚）	0pF	12KΩ
4MHz 谐振器（2 脚）	10pF	12KΩ
3.58MHz 晶振	0pF	10KΩ
3.58MHz 谐振器（2 脚）	25pF	10KΩ
2MHz 晶振和谐振器（2 脚）	25pF	10KΩ
1MHz 晶振	35pF	27KΩ
480KHz 谐振器	300pF	9.1KΩ
455KHz 谐振器	300pF	10KΩ
429KHz 谐振器	300pF	10KΩ
电阻 R1 保证了在低电压状态下，晶振被关闭。这里的低电压，是指低于 MCU 正常工作电压范围。请注意，当启动了 LVR 功能，R1 可以不接。		

指令集摘要

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(1)	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(1)	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(1)	Z,C,AC,OV
SBC	A,[m]	ACC 与数据存储器、进位标志相减，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(1)	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(1)	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(1)	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(1)	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(1)	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(1)	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(1)	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(1)	Z
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(1)	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(1)	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(1)	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(1)	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(1)	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(1)	无
SET	[m].i	置位数据存储器的位	1(1)	无

助记符		说明	指令周期	影响标志位
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(2)	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(2)	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(2)	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(2)	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(3)	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(3)	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(2)	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(2)	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(1)	无
TABRDL	[m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	2(1)	无
其它指令				
NOP		空指令	1	无
CLR	[m]	清除数据存储器	1(1)	无
SET	[m]	置位数据存储器	1(1)	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(4),PDF(4)
CLR	WDT2	预清除看门狗定时器	1	TO(4),PDF(4)
SWAP	[m]	交换数据存储器的高低字节，结果放入数据存储器	1(1)	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： 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

√：影响标志位

—：不影响标志位

(1)：如果数据是加载到 PCL 寄存器，则指令执行周期会被延长一个指令周期(四个系统时钟)。

(2)：如果满足跳跃条件，则指令执行周期会被延长一个指令周期(四个系统时钟)；否则指令执行周期不会被延长。

(3)：(1)和(2)

(4)：如果执行 CLR WDT1 或 CLR WDT2 指令后，看门狗定时器被清除，则会影响 TO 和 PDF 标志位；否则不会影响 TO 和 PDF 标志位。

ADC A, [m] 累加器与数据存储器、进位标志相加，结果放入累加器
说明： 本指令把累加器、数据存储器值以及进位标志相加，结果存放到累加器。
运算过程： ACC←ACC+[m]+C
影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADCM A, [m] 累加器与数据存储器、进位标志相加，结果放入数据存储器
说明： 本指令把累加器、数据存储器值以及进位标志相加，结果存放到存储器。
运算过程： [m]←ACC+[m]+C
影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A, [m] 累加器与数据存储器相加，结果放入累加器
说明： 本指令把累加器、数据存储器值相加，结果存放到累加器。
运算过程： ACC←ACC+[m]
影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A, x 累加器与立即数相加，结果放入累加器
说明： 本指令把累加器值和立即数相加，结果存放到累加器。
运算过程： ACC←ACC+x
影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADDM A, [m] 累加器与数据存储器相加，结果放入数据存储器
说明： 本指令把累加器、数据存储器值相加，结果存放到数据存储器。
运算过程： [m]←ACC+[m]
影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

AND A, [m] 累加器与数据存储器做“与”运算，结果放入累加器
说明： 本指令把累加器值、数据存储器值做逻辑与，结果存放到累加器。
运算过程： ACC←ACC “AND” [m]
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

AND A, x 累加器与立即数做“与”运算，结果放入累加器

说明： 本指令把累加器值、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

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

ANDM A, [m] 累加器与数据存储器做“与”运算，结果放入数据存储器

说明： 本指令把累加器值、数据存储器值做逻辑与，结果存放到数据存储器。

运算过程： ACC←ACC “AND” [m]

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CALL addr 子程序调用

说明： 本指令直接调用地址所在处的子程序，此时程序计数器加一，将此程序计数器值存到堆栈寄存器中，再将子程序所在处的地址存放到程序计数器中。

运算过程： Stack←PC+1

PC ← addr

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m] 清除数据存储器

说明： 本指令将数据存储器内的数值清零。

运算过程： [m] ←00H

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m] . i 将数据存储器的第 i 位清“0”

说明： 本指令将数据存储器内第 i 位值清零。

运算过程： [m].i ←0

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR WDT 清除看门狗定时器

说明： 本指令清除 WDT 计数器(从 0 开始重新计数)，暂停标志位(PDF)和看门狗溢出标志位(TO)也被清零。

运算过程： WDT ← 00H

PDF&TO ← 0

影响标志位

TO	PDF	OV	Z	AC	C
0	0	—	—	—	—

CLR WDT1 预清除看门狗定时器

说明： 必须搭配 CLR WDT2 一起使用，才可清除 WDT 计时器(从 0 开始重新计数)。当程序只执行过该指令，没有执行 CLR WDT2 时，系统只会不会将暂停标志位(PDF)和计数溢出位(TO)清零，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	OV	Z	AC	C
0*	0*	—	—	—	—

CLR WDT2 预清除看门狗定时器

说明： 必须搭配 CLR WDT1 一起使用，才可清除 WDT 计时器(从 0 开始重新计数)。当程序只执行过该指令，没有执行 CLR WDT1 时，系统只会不会将暂停标志位(PDF)和计数溢出位(TO)清零，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	OV	Z	AC	C
0*	0*	—	—	—	—

CPL [m] 对数据存储器取反，结果放入数据存储器

说明： 本指令是将数据存储器内保存的数值取反。

运算过程： [m] ← $[\overline{m}]$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CPLA [m] 对数据存储器取反，结果放入累加器

说明： 本指令是将数据存储器内保存的值取反后，结果存放在累加器中。

运算过程： ACC ← $[\overline{m}]$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DAA [m] 将加法运算后放入累加器的值调整为十进制数，并将结果放入数据存储器

说明 本指令将累加器高低四位分别调整为 BCD 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”，并且内部进位标志 AC1= $\overline{AC}$ ，即 AC 求反；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”再加 AC1，并把 C 置位；否则 BCD 调整就执行对原值加 AC1，C 的值保持不变。结果存放到数据存储器中，只有进位标志位(C)受影响。

操作 如果 ACC.3~ACC.0 > 9 或 AC=1

那么 [m].3~[m].0 ←(ACC.3~ACC.0)+6，AC1= $\overline{AC}$

否则 [m].3~[m].0 ←(ACC.3~ACC.0)，AC1=0

并且

如果 ACC.7~ACC.4+AC1 > 9 或 C=1

那么 [m].7~[m].4 ←(ACC.7~ACC.4)+6+ AC1，C=1

否则 [m].7~[m].4 ←(ACC.7~ACC.4)+ AC1，C=C

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

DEC [m] 数据存储器的内容减 1，结果放入数据存储器

说明： 本指令将数据存储器内的数值减一再放回数据存储器。

运算过程： [m] ← [m]-1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DECA [m] 数据存储器的内容减 1，结果放入累加器

说明： 本指令将存储器内的数值减一，再放到累加器。

运算过程： ACC ← [m]-1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

HALT 进入暂停模式

说明： 本指令终止程序执行并关掉系统时钟，RAM 和寄存器内的数值保持原状态，WDT 计数器清“0”，暂停标志位(PDF)被设为 1， WDT 计数溢出位(TO)被清为 0。

运算过程： PC ← PC+1

PDF ← 1

TO ← 0

影响标志位

TO	PDF	OV	Z	AC	C
0	1	—	—	—	—

INC [m] 数据存储器的内容加 1，结果放入数据存储器
说明： 本指令将数据存储器内的数值加一，结果放回数据存储器。
运算过程： [m] ← [m]+1
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

INCA [m] 数据存储器的内容加 1，结果放入数据存储器
说明： 本指令是将存储器内的数值加一，结果放到累加器。
运算过程： ACC ← [m]+1
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

JMP addr 无条件跳转
说明： 本指令是将要跳到的目的地直接放到程序计数器内。
运算过程： PC ← addr
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A, [m] 将数据存储器送至累加器
说明： 本指令是将数据存储器内的数值送到累加器内。
运算过程： ACC ← [m]
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A, x 将立即数送至累加器
说明： 本指令是将立即数送到累加器内。
运算过程： ACC ← x
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV [m], A 将累加器送至数据存储器
说明： 本指令是将累加器值送到数据存储器内。
运算过程： [m] ← ACC
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

NOP 空指令

说明： 本指令不作任何运算，而只将程序计数器加一。

运算过程： PC ← PC+1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

OR A, [m] 累加器与数据存储器做“或”运算，结果放入累加器

说明： 本指令是把累加器、数据存储器值做逻辑或，结果放到累加器。

运算过程： ACC←ACC “OR” [m]

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

OR A, x 累加器与立即数做“或”运算，结果放入累加器

说明： 本指令是把累加器值、立即数做逻辑或，结果放到累加器。

运算过程： ACC←ACC “OR” x

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

ORM A, [m] 累加器与数据存储器做“或”运算，结果放入数据存储器

说明： 本指令是把累加器值、存储器值做逻辑或，结果放到数据存储器。

运算过程： ACC←ACC “OR” [m]

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

RET 从子程序返回

说明： 本指令是将堆栈寄存器中的程序计数器值送回程序计数器。

运算过程： PC ← Stack

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RET A, x 从子程序返回，并将立即数放入累加器

说明： 本指令是将堆栈寄存器中的程序计数器值送回程序计数器，并将立即数送回累加器。

运算过程： PC ← Stack

ACC ← x

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RETI 从中断返回

说明： 本指令是将堆栈寄存器中的程序计数器值送回程序计数器，与 RET 不同的是它使用在中断程序结束返回时，它还会将中断控制寄存器 INTC 的 0 位(EMI)中断允许位置 1，允许中断服务。

运算过程： PC ← Stack
EMI ← 1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RL [m] 数据存储器左移一位，结果放入数据存储器

说明： 本指令是将数据存储器内的数值左移一位，第 7 位移到第 0 位，结果送回数据存储器。

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

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RLA [m] 数据存储器左移一位，结果放入累加器

说明： 本指令是将存储器内的数值左移一位，第 7 位移到第 0 位，结果送到累加器，而数据存储器内的数值不变。

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

影响标志位

TO	PDF	OV	Z	AC	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

说明： 本指令是将存储器内的数值与进位标志左移一位，第 7 位取代进位标志，进位标志移到第 0 位，结果送回数据存储器。

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

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RLCA [m] 带进位将数据存储器左移一位，结果放入累加器

说明： 本指令是将存储器内的数值与进位标志左移一位，第七位取代进位标志，进位标志移到第 0 位，结果送回累加器。

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

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RR [m] 数据存储器右移一位，结果放入数据存储器

说明： 本指令是将存储器内的数值循环右移，第 0 位移到第 7 位，结果送回数据存储器。

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

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RRA [m] 数据存储器右移一位，结果放入累加器

说明： 本指令是将数据存储器内的数值循环右移，第 0 位移到第 7 位，结果送回累加器，而数据存储器内的数值不变。

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

影响标志位

TO	PDF	OV	Z	AC	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

说明： 本指令是将存储器内的数值加进位标志循环右移，第 0 位取代进位标志，进位标志移到第 7 位，结果送回存储器。

运算过程： [m].i ←[m].(i+1); (i=0~6)

[m]. 7 ←C

C ← [m].0

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RRCA [m] 带进位将数据存储器右移一位，结果放入累加器

说明： 本指令是将数据存储器内的数值加进位标志循环右移，第 0 位取代进位标志，进位标志移到第 7 位，结果送回累加器，数据存储器内的数值不变。

运算过程： ACC.i ←[m].(i+1); (i=0~6)

ACC.7 ←C

C ← [m].0

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

SBC A,[m] 累加器与数据存储器、进位标志相减，结果放入累加器

说明： 本指令是把累加器值减去数据存储器值以及进位标志的取反，结果放到累加器。

运算过程： ACC←ACC+[$\overline{m}$]+ C

影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SBCM A,[m] 累加器与数据存储器、进位标志相减，结果放入数据存储器

说明： 本指令是把累加器值减去数据存储器值以及进位标志取反，结果放到数据存储器。

运算过程： [m]←ACC+[$\overline{m}$]+C

影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SDZ [m] 数据存储器减 1，如果结果为“0”，则跳过下一条指令

说明： 本指令是把数据存储器内的数值减 1，判断是否为 0，若为 0 则跳过下一条指令，即如果结果为零，放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程： 如果[m]-1=0，跳过下一条指令执行再下一条。

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SDZA [m] 数据存储器减 1，将结果放入累加器，如果结果为“0”，则跳过下一条指令

说明： 本指令是把数据存储器内的数值减 1，判断是否为 0，为 0 则跳过下一行指令并将减完后数据存储器内的数值送到累加器,而数据存储器内的值不变，即若结果为 0，放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程： 如果[m]-1=0，跳过下一条指令执行再下一条。

ACC ←([m]-1)

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m] 置位数据存储器

说明： 本指令是把存储器内的数值每个位置为 1。

运算过程： [m] ← FFH

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m]. i 将数据存储器的第 i 位置“1”

说明： 本指令是把存储器内的数值的第 i 位置为 1。

运算过程： [m].i ←1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZ [m] 数据存储器加 1，如果结果为“0”，则跳过下一条指令

说明： 本指令是把数据存储器内的数值加 1，判断是否为 0。若为 0，跳过下一条指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程： 如果 ([m]+1=0)，跳过下一行指令；[m] ←[m]+1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZA 数据存储器加 1，将结果放入累加器，如果结果为“0”，则跳过下一条指令

说明： 本指令是把数据存储器内的数值加 1，判断是否为 0，若为 0 跳过下一条指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)，并将加完后存储器内的数值送到累加器，而数据存储器的值保持不变。否则执行下一条指令(一个指令周期)。

运算过程： 如果[m]+1=0，跳过下一行指令；ACC ←([m]+1)

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SNZ [m]. i 如果数据存储器的第 i 位不为“0”，则跳过下一条指令

说明： 本指令是判断数据存储器内的数值的第 i 位，若不为 0，则程序计数器再加 1，跳过下一行指令，放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程： 如果 [m].i≠0，跳过下一行指令。

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SUB A, [m] 累加器与数据存储器相减，结果放入累加器

说明： 本指令是把累加器值、数据存储器值相减，结果放到累加器。

运算过程： ACC←ACC+[$\overline{m}$]+1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUB A, x 累加器与立即数相减，结果放入累加器

说明： 本指令是把累加器值、立即数相减，结果放到累加器。

运算过程： ACC←ACC+$\overline{x}$+1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUBM A, [m] 累加器与数据存储器相减，结果放入数据存储器

说明： 本指令是把累加器值、存储器值相减，结果放到存储器。

运算过程： [m]←ACC+[$\overline{m}$]+1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SWAP [m] 交换数据存储器的高低字节，结果放入数据存储器

说明： 本指令是将数据存储器的低四位和高四位互换，再将结果送回数据存储器。

运算过程： [m].7~[m].4↔[m].3~[m].0

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SWAPA [m] 交换数据存储器的高低字节，结果放入累加器

说明： 本指令是将数据存储器的低四位和高四位互换，再将结果送回累加器。

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

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m] 如果数据存储器为“0”，则跳过下一条指令

说明： 本指令是判断数据存储器内的数值是否为 0，为 0 则跳过下一行指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程： 如果 [m] = 0, 跳过下一行指令。

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZA [m] 数据存储器送至累加器，如果内容为“0”，则跳过下一条指令

说明： 本指令是判断存储器内的数值是否为 0，若为 0 则跳过下一行指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以得正确的指令(二个指令周期)。并把存储器内值送到累加器，而存储器的值保持不变。否则执行下一条指令(一个指令周期)。

运算过程： 如果[m] = 0，跳过下一行指令，并 ACC← [m]。

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m]. i 如果数据存储器的第 i 位为“0”，则跳过下一条指令
说明： 本指令是判断存储器内第 i 位值是否为 0，若为 0 则跳过下一行指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。
运算过程： 如果 [m].i = 0，跳过下一行指令。
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDC [m] 读取 ROM 当前页的内容，并送至数据存储器和 TBLH
说明： 本指令是将表格指针指向程序寄存器当前页，将低位送到存储器，高位直接送到 TBLH 寄存器内。
运算过程： [m] ←程序存储器低四位
TBLH←程序存储器高四位
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDL [m] 读取 ROM 最后一页的内容，并送至数据存储器和 TBLH
说明： 本指令是将 TABLE 指针指向程序寄存器最后页，将低位送到存储器，高位直接送到 TBLH 寄存器内。
运算过程： [m] ←程序存储器低四位
TBLH←程序存储器高四位
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

XOR A, [m] 累加器与立即数做“异或”运算，结果放入累加器
说明： 本指令是把累加器值、数据存储器值做逻辑异或，结果放到累加器。
运算过程： ACC←ACC “XOR” [m]
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XORM A, [m] 累加器与数据存储器做“异或”运算，结果放入数据存储器
说明： 本指令是把累加器值、数据存储器值做逻辑异或，结果放到数据存储器。
运算过程： [m]←ACC “XOR” [m]
影响标志位

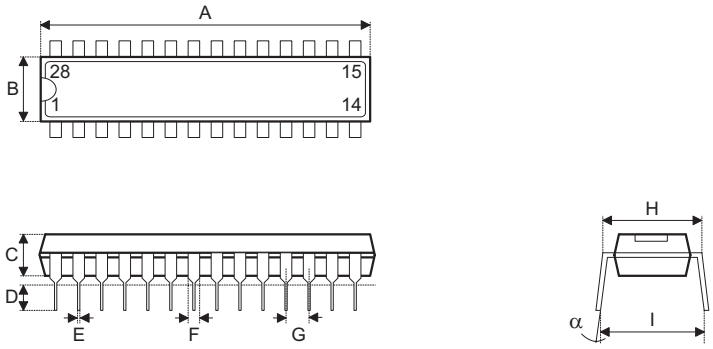
TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XOR A, x 累加器与数据存储器做“异或”运算，结果放入累加器
说明： 本指令是把累加器值与立即数做逻辑异或，结果放到累加器。
运算过程： ACC←ACC “XOR” x
影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

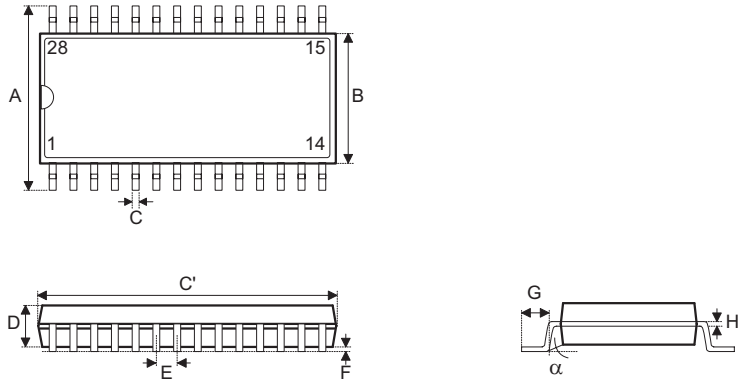
封装信息

28-pin SKDIP (300mil)外形尺寸



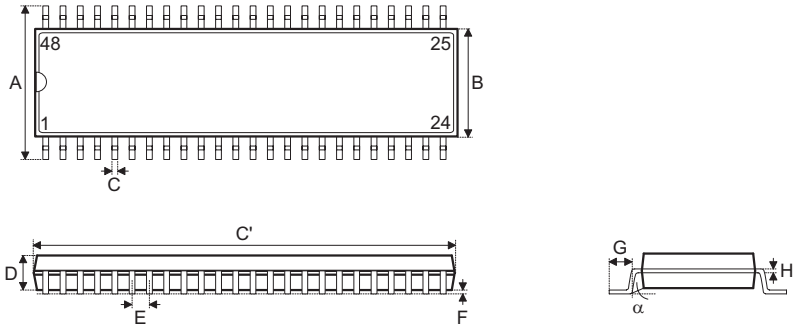
标号	尺寸（mil）		
	Min.	**Nom.**	**Max.**
A	1375	--	1395
B	278	--	298
C	125	--	135
D	125	--	145
E	16	--	20
F	50		70
G	--	100	--
H	295	--	315
I	330	--	375
α	0°	--	15°

28-pin SOP (300mil)外形尺寸



标号	尺寸（mil）		
	Min.	**Nom.**	**Max.**
A	394	--	419
B	290	--	300
C	14	--	20
C’	697	--	713
D	92	--	104
E	--	50	--
F	4	--	--
G	32	--	38
H	4	--	12
α	0°	--	10°

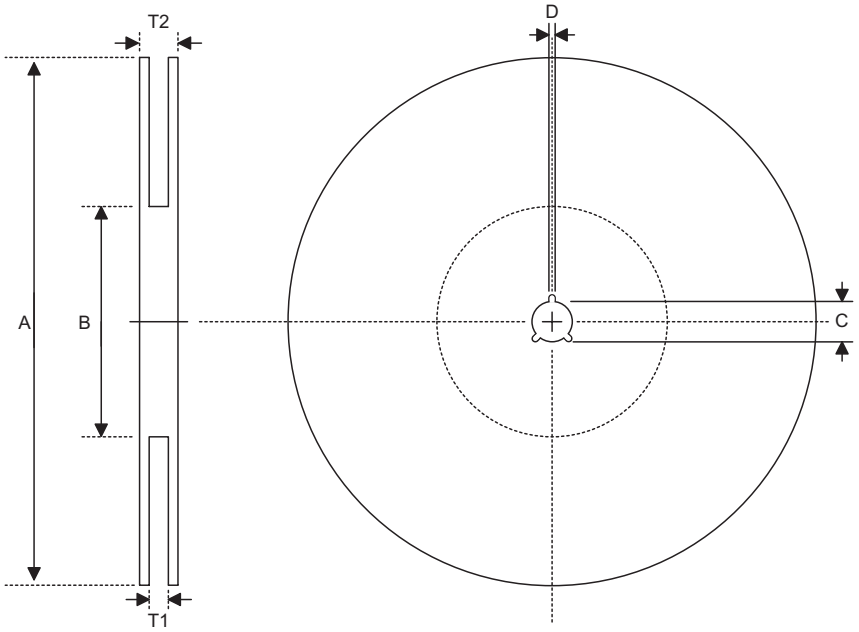
48-pin SSOP (300mil)外形尺寸



标号	尺寸（mil）		
	Min.	**Nom.**	**Max.**
A	395	--	420
B	291	--	299
C	8	--	12
C’	613	--	637
D	85	--	99
E	--	25	--
F	4	--	10
G	25	--	35
H	4	--	12
α	0°	--	8°

包装带和卷轴规格：

卷轴尺寸：



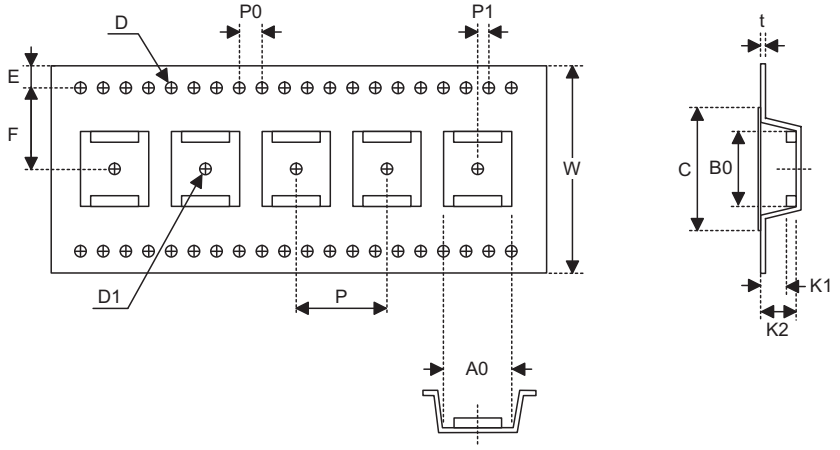
SOP 28W(300mil)

标号	描述	尺寸(mm)
A	卷轴外圈直径	330±1.0
B	卷轴内圈直径	62±1.5
C	轴心直径	13.0+0.5 -0.2
D	缝宽	2.0±0.5
T1	轮缘宽	24.8+0.3 -0.2
T2	卷轴宽	30.2±0.2

SSOP 48W

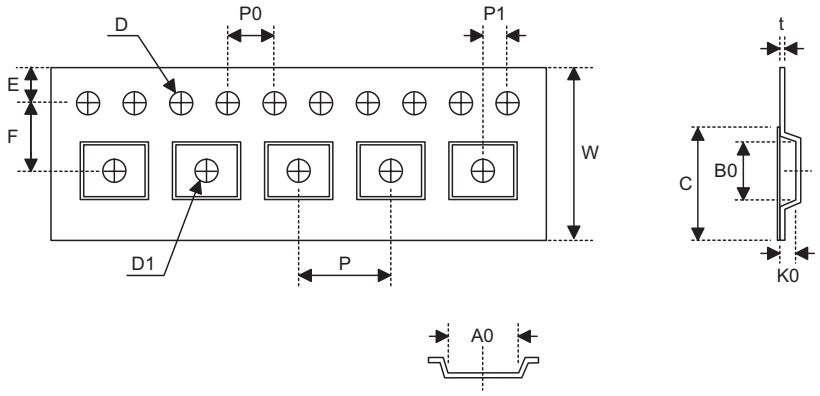
标号	描述	尺寸(mm)
A	卷轴外圈直径	330±1.0
B	卷轴内圈直径	100±0.1
C	轴心直径	13.0+0.5 -0.2
D	缝宽	2.0±0.5
T1	轮缘宽	32.2+0.3 -0.2
T2	卷轴宽	38.2±0.2

运输带尺寸：



S0P 28W(300mil)

标号	描述	尺寸(mm)
W	运输带宽	24.0±0.3
P	空穴间距	12.0±0.1
E	穿孔位置	1.75±0.1
F	空穴至穿孔距离（宽度）	11.5±0.1
D	穿孔直径	1.5+0.1
D1	空穴中之小孔直径	1.5+0.25
P0	穿孔间距	4.0±0.1
P1	空穴至穿孔距离（长度）	2.0±0.1
A0	空穴长	10.85±0.1
B0	空穴宽	18.34±0.1
K0	空穴深	2.97±0.1
t	传输带厚度	0.35±0.01
C	覆盖带宽度	21.3



SS0P 48W

标号	描述	尺寸(mm)
W	运输带宽	32.0±0.3
P	空穴间距	16.0±0.1
E	穿孔位置	1.75±0.1
F	空穴至穿孔距离（宽度）	14.2±0.1
D	穿孔直径	2.0 Min.
D1	空穴中之小孔直径	1.5+0.25
P0	穿孔间距	4.0±0.1
P1	空穴至穿孔距离（长度）	2.0±0.1
A0	空穴长	12.0±0.1
B0	空穴宽	16.20±0.1
K1	空穴深	2.4±0.1
K2	空穴深	3.2±0.1
t	传输带厚度	0.35±0.05
C	覆盖带宽度	25.5

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 889 号 2 号楼 7 楼 200233
电话: 021-6485-5560
传真: 021-6485-0313
网站: www.holtek.com.cn

盛扬半导体有限公司（深圳业务处）
深圳市深南中路赛格广场 43 楼 518031
电话: 0755-8346-5589
传真: 0755-8346-5590
ISDN: 0755-834-65591

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

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: www.holmate.com