

技术相关信息

- [工具信息](#)
- [FAQs](#)
- [应用范例](#)
 - [HA0003S HT48 & HT46 MCU 与 HT93LC46 EEPROM 的通信](#)
 - [HA0004S HT48 & HT46 MCU UART 的软件实现方法](#)
 - [HA0005S HT48 & HT46 MCU 用软件执行 I²C 总线的控制功能的方法](#)
 - [HA0047S HT46 MCU 的 PWM 的应用范例](#)
 - [HA0075S MCU 重置电路及振荡电路应用](#)

特性

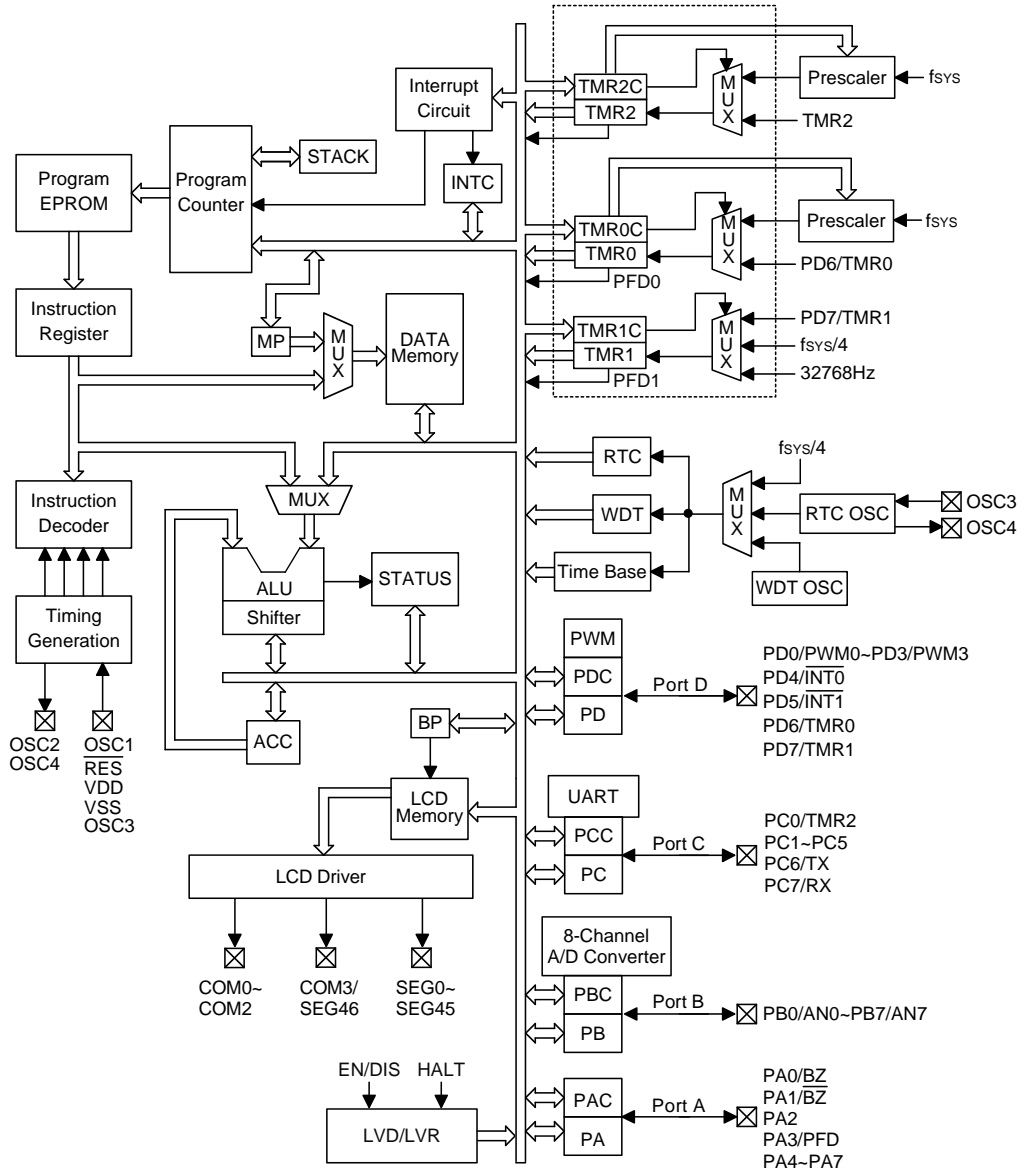
- 工作电压：
 - $f_{\text{SYS}}=4\text{MHz}$: 2.2V~5.5V
 - $f_{\text{SYS}}=8\text{MHz}$: 3.3V~5.5V
- 32 个双向输入/输出口
- 2 个外部中断输入
- 2 个 16 位定时/计数器, 具有 PFD(可编程分频器)功能
- 1 个 8 位定时/计数器, 可编程分频器
- 47×3 或 46×4 段的 LCD 驱动(SEG0~SEG23 可由配置选项设置为逻辑输出)
- 16K×16 程序存储器
- 576×8 数据存储器
- 通用异步接收/发送器 (UART)
- 具有 PFD 功能, 可用于发声
- 1 个实时时钟(RTC)
- 1 个 8 位的实时时钟预分频器
- 看门狗定时器
- 蜂鸣器输出
- 内置晶体、RC 和 32768Hz 晶体振荡电路
- HALT 和唤醒功能可降低功耗
- 16 层硬件堆栈
- 8 通道 12 位解析度的 A/D 转换器
- 4 通道 PWM 输出, 与 4 个输入/输出口共用引脚
- 位操作指令
- 查表指令, 表格内容字长 16 位
- 系统频率为 8MHz 时, 指令周期为 0.5 μ s
- 63 条指令
- 指令执行时间为 1 或 2 个指令周期
- 低电压复位/检测功能
- 52-pin QFP, 56-pin SSOP, 100-pin LQFP 封装

概述

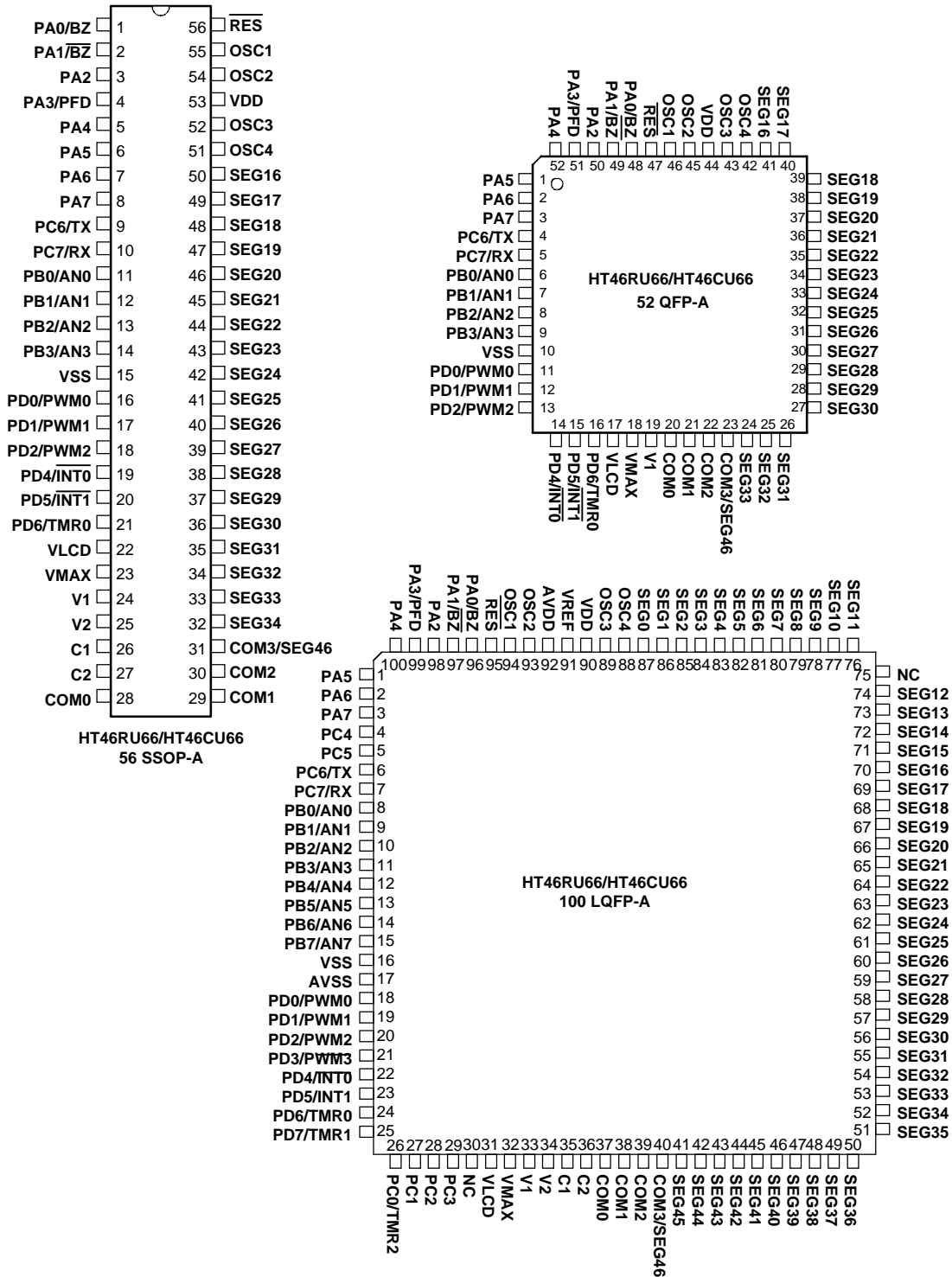
HT46RU66/HT46CU66 是 8 位高性能精简指令集单片机, 专门为需要 A/D 转换和 LCD 显示的产品而设计。配置版本 HT46CU66 与 OTP 版本 HT46RU66 引脚和功能完全相同。

低功耗、I/O 使用灵活、计数器、振荡类型选择、多通道 A/D 转换、脉冲测量功能、暂停和唤醒功能, 以及 LCD 显示功能, 使这款单片机可以广泛应用于需要 A/D 转换和 LCD 显示的产品中, 例如电子测量仪器、环境监控、手持式测量工具、马达控制等工业和家庭系统中。

方框图



引脚图



引脚说明

引脚名称	输入/输出	配置选项	说明
PA0/BZ PA1/ \overline{BZ} PA2 PA3/PFD PA4~PA7	输入/输出	唤醒功能 上拉电阻 蜂鸣器 PFD	8 位双向输入/输出口。每一位可由配置选项设置为唤醒输入。可由软件设置为 CMOS 输出、带或不带上拉电阻(由上拉电阻选项决定: 位选择)的斯密特触发输入。BZ、 \overline{BZ} 和 PFD 分别与 PA0、PA1 和 PA3 共用引脚。
PB0/AN0~ PB7/AN7	输入/输出	上拉电阻	8 位双向输入/输出口。可由软件设置为 CMOS 输出、带或不带上拉电阻(由上拉电阻选项决定: 位选择)的斯密特触发输入或 A/D 输入。一旦 PB 口做为 A/D 输入(由软件设置), 则其输入/输出功能和上拉电阻会自动失效。
PC0/TMR2 PC1~PC5 PC6/TX PC7/RX	输入/输出	上拉电阻	8 位双向输入/输出口。每一位可由配置选项设置为唤醒输入。可由软件设置为 CMOS 输出、带或不带上拉电阻(由上拉电阻选项决定: 位选择)的斯密特触发输入。TMR2、TX 和 RX 分别与 PC0、PC6 和 PC7 共用引脚。
PD0/PWM0 PD1/PWM1 PD2/PWM2 PD3/PWM3 PD4/ $\overline{INT0}$ PD5/ $\overline{INT1}$ PD6/TMR0 PD7/TMR1	输入/输出	上拉电阻 PWM 中断	8 位双向输入/输出口。可由软件设置为 CMOS 输出、带或不带上拉电阻(由上拉电阻选项决定: 位选择)的斯密特触发输入。PWM0/PWM1/PWM2/PWM3 输出与 PD0/PD1/PD2/PD3 共用引脚(由 PWM 选项决定)。 $\overline{INT0}$ 和 $\overline{INT1}$ 与 PD4 和 PD5 共用引脚, 配置选项决定使能/禁止中断以及设置上升沿/下降沿触发中断。TMR0 和 TMR1 分别与 PD6 和 PD7 共用引脚。
OSC1 OSC2	输入 输出	晶体或 RC	OSC1、OSC2 连接 RC 或晶体(由配置选项确定)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。系统时钟也可以选择为 RTC 振荡, 如果选择 RTC 振荡作为系统时钟, 则这两个引脚可以空接。
OSC3 OSC4	输入 输出	RTC 或系统时钟	实时时钟振荡器。OSC3、OSC4 连接 32768Hz 的晶体振荡器, 用于提供实时时钟或系统时钟(由配置选项确定)。
VLCD	输入	—	LCD 电源。
VMAX	输入	—	IC 最高电压, 接至 VDD、VLCD 或 V1。
V1,V2,C1,C2	输入	—	电压泵。
SEG0~SEG7	输出	SEG0~SEG7 或 逻辑输出	LCD 驱动的 Segment 输出。SEG0~SEG7 可配置选择为逻辑输出口。
SEG8~SEG15	输出	SEG8~SEG15 或 逻辑输出	LCD 驱动的 Segment 输出。SEG8~SEG15 可配置选择为逻辑输出口。
SEG16~SEG23	输出	SEG16~SEG23 或 逻辑输出	LCD 驱动的 Segment 输出。SEG16~SEG23 每一位可配置选择为逻辑输出口。
SEG24~SEG45	输出	—	LCD 驱动的 Segment 输出。
COM0~COM2 COM3 / SEG46	输出	1/3 或 1/4 Duty COM3 或 SEG46	SEG46 可由配置选项设置为 LCD 显示的 Segment 或 Common 输出端。COM0~COM2 是 LCD 驱动的 Common 输出。
VREF	输入	—	参考电压输入。

$\overline{\text{RES}}$	输入	—	斯密特触发复位输入。
VDD	—	—	正电源。
VSS	—	—	负电源，接地。
AVDD	输入	—	模拟电压正电源输入。
AVSS	输入	—	模拟电压负电源输入，接地。

注意： PA 上的每个引脚可通过配置选项被设定成具有唤醒功能。

引脚可以单独的选择带上拉电阻。

引脚 V2、C1、C2 和 SEG34 在 52-pin 的 QFP 封装中没有。

引脚 PB4/AN4 和 PB7/AN7 只有在 100-pin 的 QFP 封装中才有。

引脚 PC0-PC5 只有在 100-pin 的 QFP 封装中才有。

引脚 PD3/PWM3 只有在 100-pin 的 QFP 封装中才有。

引脚 PD7/TMR1 和 PC0/TMR2 只有在 100-pin 的 QFP 封装中才有，而在 56-pin 的 SSOP 和 52-pin QFP 封装中只有引脚 TMR0。

Segment 引脚 SEG0~SEG15 和 SEG35~SEG45 只有在 100-pin 的 QFP 封装中才有。

对于 56-pin 的 SSOP 和 52-pin QFP 封装，VREF、AVDD 是和 VDD 连接在一起。

对于 56-pin 的 SSOP 和 52-pin QFP 封装，AVSS 和 VSS 连接在一起。

极限参数

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

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°C

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} =4MHz	2.2	—	5.5	V
		—	f _{SYS} =8MHz	3.3	—	5.5	V
AV _{DD}	模拟电路工作电压 (见注释 5)	—	V _{REF} =AV _{DD}	3.0	—	5.5	V
I _{DD1}	工作电流 (晶体振荡、RC 振荡)	3V	无负载, f _{SYS} =4MHz, ADC 关闭, UART 关闭	—	1	2	mA
		5V	ADC 关闭, UART 关闭	—	3	5	mA
I _{DD2}	工作电流 (晶体振荡, RC 振荡)	3V	无负载, f _{SYS} =4MHz, ADC 关闭, UART 打开	—	1.5	3	mA
		5V	ADC 关闭, UART 打开	—	3	6	mA
I _{DD3}	工作电流 (晶体振荡, RC 振荡)	5V	无负载, f _{SYS} =8MHz, ADC 关闭, UART 关闭	—	4	8	mA
I _{DD4}	工作电流 (晶体振荡, RC 振荡)	5V	无负载, f _{SYS} =8MHz, ADC 关闭, UART 打开	—	5	10	mA
I _{DD5}	工作电流 (f _{SYS} =32768Hz)	3V	无负载, ADC 关闭, UART 关闭	—	0.3	0.6	mA
		5V	UART 关闭	—	0.6	1	mA
I _{STB1}	静态电流 (*f _s =T1)	3V	无负载, 系统 HALT, HALT 时 LCD 关闭, UART 关闭	—	—	1	μA
		5V	UART 关闭	—	—	2	μA
I _{STB2}	静态电流 (*f _s =RTC 振荡)	3V	无负载, 系统 HALT, HALT 时 LCD 打开 电容型偏压, UART 关闭	—	2.5	5	μA
		5V	电容型偏压, UART 关闭	—	10	20	μA
I _{STB3}	静态电流 (*f _s =WDT 振荡)	3V	无负载, 系统 HALT, HALT 时 LCD 打开, 电容型偏压, UART 关闭	—	2	5	μA
		5V	电容型偏压, UART 关闭	—	6	10	μA
I _{STB4}	静态电流 (*f _s =RTC 振荡)	3V	无负载, 系统 HALT, HALT 时 LCD 打开, 电阻型偏压, 1/2bias, V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	17	30	μA
		5V	V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	34	60	μA
I _{STB5}	静态电流 (*f _s =RTC 振荡)	3V	无负载, 系统 HALT, HALT 时 LCD 打开, 电阻型偏压, 1/3bias, V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	13	25	μA
		5V	V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	28	50	μA
I _{STB6}	静态电流 (*f _s =WDT 振荡)	3V	无负载, 系统 HALT, HALT 时 LCD 打开, 电阻型偏压, 1/2bias, V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	14	25	μA
		5V	V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	26	50	μA
I _{STB7}	静态电流 (*f _s =WDT 振荡)	3V	无负载, 系统 HALT, HALT 时 LCD 打开, 电阻型偏压, 1/3bias, V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	10	20	μA
		5V	V _{LCD} =V _{DD} , UART 关闭 (选择低电流偏压)	—	19	40	μA

V_{IL1}	输入/输出、TMR0、TMR1、 $\overline{INT0}$ 、 $\overline{INT1}$ 的低电平输入电压	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	输入/输出、TMR0、TMR1、 $\overline{INT0}$ 、 $\overline{INT1}$ 的低电平输入电压	—	—	$0.7V_{DD}$	—	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	—	—	2.7	3.0	3.3	V
V_{LVD}	低电压检测	—	—	3.0	3.3	3.6	V
V_{AD}	AD 输入	—	52QFP, 56SSOP	0	—	A_{VDD}	V
			100QFP	0	—	V_{REF}	V
V_{REF}	AD 转换器参考电压范围	—	$A_{VDD}=3V$	1.3	—	A_{VDD}	V
			$A_{VDD}=5V$	1.5	—	A_{VDD}	V
I_{OL1}	输入/输出逻辑输出灌电流	3V	$V_{OL}=0.1V_{DD}$	6	12	—	mA
		5V		10	25	—	mA
I_{OH1}	输入/输出逻辑输出源电流	3V	$V_{OH}=0.9V_{DD}$	-2	-4	—	mA
		5V		-5	-8	—	mA
I_{OL2}	LCD Com 和 Segment 电流	3V	$V_{OL}=0.1V_{DD}$	210	420	—	μA
		5V		350	700	—	μA
I_{OH2}	LCD Com 和 Segment 电流	3V	$V_{OH}=0.9V_{DD}$	-80	-160	—	μA
		5V		-180	-360	—	μA
R_{PH}	输入/输出、 $\overline{INT0}$ 和 $\overline{INT1}$ 上拉电阻	3V	—	20	60	100	k Ω
		5V	—	10	30	50	k Ω
I_{ADC}	A/D 转换电路打开时增加的系统功耗	3V	$t_{AD}=1\mu s$	—	0.5	1	mA
		5V		—	1.5	3	mA
DNL	AD 转换器微分非线性	5V	$A_{VDD}=5V, V_{REF}=A_{VDD}$ $t_{AD}=1\mu s$	—	—	± 2	LSB
INL	AD 转换器积分非线性	5V	$A_{VDD}=5V, V_{REF}=A_{VDD}$ $t_{AD}=1\mu s$	—	± 2.5	± 4	LSB
RESOLU	分辨率	—	—	—	—	12	Bits

注:

1. 有关“ f_s ”的具体说明请参阅 WDT 的时钟选择。
2. $I_{STB1}=WDT$ 关闭, $I_{STB2}\sim I_{STB7}=WDT$ 打开
3. A_{VDD} 与 V_{DD} 电压必须相同

交流电气特性

Ta=25°C

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, RC 振荡)	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	
f _{SYS2}	系统时钟 (32768Hz 晶体振荡)	—	2.2V~5.5V	—	32768	—	Hz
f _{RTCOSC}	RTC 频率	—	—	—	32768	—	Hz
f _{TIMER}	定时器输入频率 (TMR0/TMR1)	—	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 _{RES}	外部复位低电平脉宽	—	—	1	—	—	μs
t _{SST}	系统启动延迟时间	—	上电或从 HALT 状态唤醒	—	1024	—	t _{SYS}
t _{INT}	中断脉冲宽度	—	—	1	—	—	μs
t _{AD}	A/D 时钟周期	—	—	1	—	—	μs
t _{ADC}	A/D 转换时间	—	—	—	80	—	t _{AD}
t _{ADCS}	A/D 采样时间	—	—	—	32	—	t _{AD}

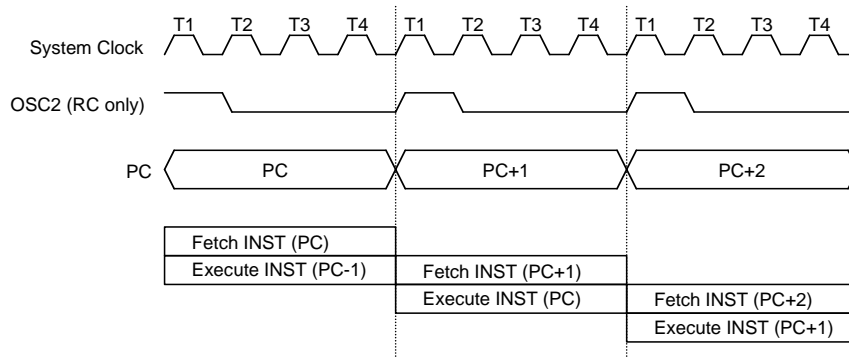
 注: t_{SYS} = 1/f_{SYS1} 或 1/f_{SYS2}

系统功能说明

指令执行时序

单片机的系统时钟由晶体振荡器或 RC 振荡器或 32768Hz 的晶体振荡器产生。该时钟在芯片内部被分成四个互不重叠的时钟周期。一个指令周期包括四个系统时钟周期。

指令的读取和执行是以流水线方式进行的，这种方式在一个指令周期进行读取指令操作，而在下一个指令周期进行解码与执行该指令。因此，流水线方式使多数指令能在一个周期内执行完成。但如果涉及到的指令要改变程序计数器的值，就需要花两个指令周期来完成这一条指令。



指令执行时序

程序计数器 — PC

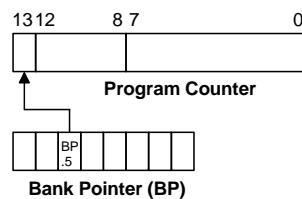
14 位的程序计数器(PC)控制程序存储器 ROM 中指令执行的顺序，它可寻址整个 ROM 范围的 16384 (×16 位) 个地址。

取得指令码以后，程序计数器会自动加一，指向下一个指令码的地址。但如果执行跳转、条件跳转、向 PCL(程序计数器低字节寄存器)赋值、子程序调用、初始化复位、中断或子程序返回等操作时，PC 会载入与指令相关的地址而非下一条指令地址。

当遇到条件跳跃指令且符合条件时，当前指令执行过程中读取的下一条指令会被丢弃，取而代之的是一个空指令周期，随后才能取得正确的指令。反之，就会顺序执行下一条指令。

程序计数器的低字节(PCL)是一个可读写的寄存器(06H)。对 PCL 赋值将产生一个短跳转动作，跳转的范围为当前页 256 个地址。当遇到控制转移指令时，系统也会插入一个空指令周期。

当程序存储在两个 BANK 中，可通过 BP 第 5 位 (程序计数器 PC 最高位) 选择正确的 BANK。



模式	程序计数器													
	*13	*12	*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	0	0
外部中断 0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
外部中断 1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
定时/计数器 0 溢出	0	0	0	0	0	0	0	0	0	0	1	1	0	0
定时/计数器 1 溢出	0	0	0	0	0	0	0	0	0	1	0	0	0	0
UART 中断	0	0	0	0	0	0	0	0	0	1	0	1	0	0
多功能中断	0	0	0	0	0	0	0	0	0	1	1	0	0	0
条件跳转	PC+2 (包括当前 bank)													
装载 PCL	*13	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
跳转, 子程序调用	BP.5	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
子程序返回 (RET、RETI)	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

程序计数器

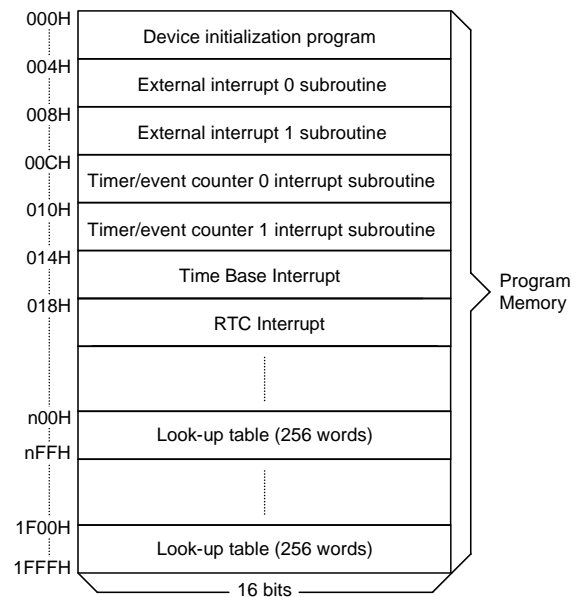
注: *13~*0 : 程序计数器位 S13~S0 : 堆栈寄存器位
 #13~#0 : 指令代码位 @7~@0 : PCL 位

程序存储器

程序存储器用来存放要执行的指令代码, 以及一些数据、表格和中断入口。程序存储器有 16384×16 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。程序存储器空间被分为两个 Bank, Bank0 和 Bank1。每个 Bank 空间容量为 8192×16, 可通过 Bank 指针 BP.5 选择。当 BP=000XXXXXB 时, 选择 Bank0; 当 BP=001XXXXXB 时, 选择 Bank1。指令 JMP 和 CALL 只包含 13 位地址, 其跳转范围为一个 bank (即 8K)。当执行 JMP 或 CALL 指令, 必须设置 Bank 指针 BP.5, 以确保程序正确执行。若从中断或子程序返回时, 14 位 PC 将从堆栈弹出, 所以在执行 RET 或 RETI 指令时不需要改变 BP.5 值。

以下列出的程序存储器地址是系统专为特殊用途而保留的:

- 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 0 服务程序保留。当 $\overline{INT0}$ 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
该地址为外部中断 1 服务程序保留。当 $\overline{INT1}$ 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 008H 地址开始执行。
- 地址 00CH
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 00CH 地址开始执行。
- 地址 010H
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,



Note: n ranges from 0 to 1F

程序存储器

则程序会跳转到 010H 地址开始执行。

- 地址 014H

该地址为 UART 中断服务程序保留。当传输或接收完成触发 UART 中断，如果中断允许且堆栈未空，则程序会跳转到 014H 地址开始执行。

- 地址 018H

该地址为多功能中断服务程序保留。当定时/计数器 2、RTC 或时基触发中断，如果中断允许且堆栈未空，则程序会跳转到 018H 地址开始执行。

- 表格区

ROM 空间的任何地址都可做为查表使用。查表指令“TABRDC [m]”(页由 TBHP 指定，查当前页表格，1 页=256 个字)和“TABRDL [m]”(查最后页表格)，会把表格内容低字节传送给[m]，而表格内容高字节传送到 TBLH 寄存器(08H)。只有表格内容的低字节被传送到目标地址中，而高字节被传送到表格内容高字节寄存器 TBLH。表格内容高字节寄存器 TBLH 是只读寄存器。表格指针 TBHP(1FH)和 TBLP (07H) 是可读/写寄存器，用来指明表格地址。在查表之前，要先将表格地址写入 TBHP 和 TBLP 中。所有与表格有关的指令都需要两个指令周期的执行时间。这里提到的表格区都可以做为正常的程序存储器来使用。

指令	表格区								
	*13 ~ *8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC[m]	TBHP	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL[m]	111111	@7	@6	@5	@4	@3	@2	@1	@0

表格区

注：*13 ~ *0 : 表格地址字节

TBHP : 表格指针高位

@7 ~ @0 : 表格指针低位(TBLP)

堆栈寄存器 — STACK

堆栈寄存器是特殊的存储器空间，用来保存 Program Counter 的值。HT46xU66 有 16 层堆栈，堆栈寄存器既不是数据存储器的一部分，也不是程序存储器的一部分，而且它既不能读出，也不能写入。堆栈的使用是通过堆栈指针(SP)来实现的，堆栈指针也不能读出或写入。当发生子程序调用或中断响应时，程序计数器(Program Counter)的值会被压入堆栈；在子程序调用结束或中断响应结束时(执行指令 RET 或 RETI)，堆栈将原先压入堆栈的内容弹出，重新装入程序计数器中。在系统复位后，堆栈指针会指向堆栈顶部。

如果堆栈已满，并且发生了不可屏蔽的中断，那么只有中断请求标志会被记录下来，而中断响应会被抑制，直到堆栈指针(执行 RET 或 RETI 指令)发生递减，中断才会被响应。这个功能可以防止堆栈溢出，使得程序员易于使用这种结构。同样，如果堆栈已满，并且发生了子程序调用，那么堆栈会发生溢出，首先进入堆栈的内容将会丢失，只有最后的 16 个返回地址会被保留。

数据存储器 — RAM

数据存储器由 620×8 位组成，分为两个功能区间：特殊功能寄存器(44×8)和通用数据存储器 Bank 0: 192×8、Bank 2: 192×8、Bank 3: 192×8，数据存储器单元大多数是可读/写的，但有些是只读的。

特殊功能寄存器包括间接寻址寄存器 0(00H)，间接寻址指针寄存器 0(MP0: 01H)，间接寻址寄存器 1(02H)，间接寻址指针寄存器 1(MP1: 03H)，存储器段指针(BP: 04H)，累加器(ACC: 05H)，程序计数器低字节寄存器(PCL: 06H)，表格指针寄存器(TBLP: 07H)，表格指针高字节寄存器(TBHP: 1FH)，表格内容高字节寄存器(TBLH: 08H)，RTC 控制寄存器(RTCC: 09H)，状态寄存器 (STATUS: 0AH)，中断控制寄存器 0(INTC0: 0BH)，定时/计数器 0(TMR0H: 0CH, TMR0L: 0DH)，定时/计数器 0 控制寄存器 (TMR0C: 0EH)，定时/计数器 1(TMR1H: 0FH, TMR1L: 10H)，定时/计数器 1 控制寄存器(TMR1C: 11H)，定时/计数器 2 (TMR2: 2DH)，定时/计数器 2 控制寄存器 (TMR2C: 2EH)，中断控制寄存器 1(INTC1: 1EH)，多功能控制寄存器 (MFIC: 2FH)，PWM 数据寄存器(PWM0: 1AH, PWM1: 1BH, PWM2: 1CH, PWM3: 1DH)，A/D 转换结果低字节寄存器(ADRL: 24H)，A/D 转换结果高字节寄存器(ADRH: 25H)，A/D 控制寄存器(ADCR: 26H)，A/D 时钟设置寄存器(ACSR: 27H)，输入/输出寄存器(PA: 12H, PB: 14H, PC: 16H, PD: 18H)，输入/输出控制寄存器(PAC: 13H, PBC: 15H, PCC: 17H, PDC: 19H)，UART 状态寄存器 (USR: 30H)，UART 控制寄存器 1 (UCR1: 31H)，UART 控制寄存器 2 (UCR2: 32H)，UART 发送和接收寄存器 (TXR/RXR: 33H)，UART 波特率发生器寄存器 (BRG: 34H)。

其余在 40H 之前的空间保留给系统以后扩展使用，读取这些地址的返回值为“00H”。而在通用数据寄存器地址从 40H 到 FFH(Bank 0: BP=0、Bank 2: BP=2 或 Bank 3: BP=3)，用来存储数据和控制信息。所有的数据存储器单元都能直接执行算术、逻辑、递增、递减和循环操作。除了一些特殊位外，数据存储器的每一位都可通过“SET[m].i”置位或由“CLR[m].i”复位。而且都可以通过间接寻址指针(MP0: 01H/MP1: 03H)进行间接寻址。

00H	Indirect Addressing Register 0
01H	MP0
02H	Indirect Addressing Register 1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	RTCC
0AH	STATUS
0BH	INTC0
0CH	TMR0H
0DH	TMR0L
0EH	TMR0C
0FH	TMR1H
10H	TMR1L
11H	TMR1C
12H	PA
13H	PAC
14H	PB
15H	PBC
16H	PC
17H	PCC
18H	PD
19H	PDC
1AH	PWM0
1BH	PWM1
1CH	PWM2
1DH	PWM3
1EH	INTC1
1FH	TBHP
20H	
21H	
22H	
23H	
24H	ADRL
25H	ADRH
26H	ADCR
27H	ACSR
28H	
29H	
2AH	
2BH	
2CH	
2DH	TMR2
2EH	TMR2C
2FH	MFIC
30H	USR
31H	UCR1
32H	UCR2
33H	TXR/RXR
34H	BRG
35H	
36H	
37H	
38H	
39H	
3AH	
3BH	
3CH	
3DH	
3EH	
3FH	
40H	General Purpose Data Memory 192 Bytes × 3 (3 Banks: Bank0, Bank2, Bank3)
41H	
42H	
43H	
44H	
45H	
46H	
47H	
48H	
49H	
4AH	
4BH	
4CH	
4DH	
4EH	
4FH	
50H	
51H	
52H	
53H	
54H	
55H	
56H	
57H	
58H	
59H	
5AH	
5BH	
5CH	
5DH	
5EH	
5FH	
60H	
61H	
62H	
63H	
64H	
65H	
66H	
67H	
68H	
69H	
6AH	
6BH	
6CH	
6DH	
6EH	
6FH	
70H	
71H	
72H	
73H	
74H	
75H	
76H	
77H	
78H	
79H	
7AH	
7BH	
7CH	
7DH	
7EH	
7FH	
80H	
81H	
82H	
83H	
84H	
85H	
86H	
87H	
88H	
89H	
8AH	
8BH	
8CH	
8DH	
8EH	
8FH	
90H	
91H	
92H	
93H	
94H	
95H	
96H	
97H	
98H	
99H	
9AH	
9BH	
9CH	
9DH	
9EH	
9FH	
A0H	
A1H	
A2H	
A3H	
A4H	
A5H	
A6H	
A7H	
A8H	
A9H	
AAH	
A BH	
A CH	
A DH	
A EH	
A FH	
B0H	
B1H	
B2H	
B3H	
B4H	
B5H	
B6H	
B7H	
B8H	
B9H	
BAH	
B BH	
B CH	
B DH	
B EH	
B FH	
C0H	
C1H	
C2H	
C3H	
C4H	
C5H	
C6H	
C7H	
C8H	
C9H	
CAH	
C BH	
C CH	
C DH	
C EH	
C FH	
D0H	
D1H	
D2H	
D3H	
D4H	
D5H	
D6H	
D7H	
D8H	
D9H	
DAH	
D BH	
D CH	
D DH	
D EH	
D FH	
E0H	
E1H	
E2H	
E3H	
E4H	
E5H	
E6H	
E7H	
E8H	
E9H	
E AH	
E BH	
E CH	
E DH	
E EH	
E FH	
F0H	
F1H	
F2H	
F3H	
F4H	
F5H	
F6H	
F7H	
F8H	
F9H	
FAH	
F BH	
F CH	
F DH	
F EH	
F FH	

Special Purpose Data Memory

■ : Unused
Read as "00"

数据存储器

间接寻址寄存器

地址 00H 和 02H 是间接寻址寄存器，并无实际的物理区存在。任何对[00H]或[02H]的读/写操作，都是访问由 MP0(01H)MP1(03H)或所指向的 RAM 单元。间接读取地址 00H 或 02H 得到的值为 00H，间接写入此地址，不会产生任何操作。

间接寻址寄存器之间不支持数据传送功能。间接寻址指针 MP0 和 MP1 是 8 位寄存器。MP0 只能用于寻址数据存储器，而 MP1 能用于寻址数据存储器 and LCD 显示存储器。

累加器

累加器(ACC)与算术逻辑单元(ALU)有密切关系。它对应于 RAM 地址 05H，做为运算的立即数据。存储器之间的数据传送必须经过累加器。

算术逻辑单元 — ALU

算术逻辑单元(ALU)是执行 8 位算术、逻辑运算的电路，它提供有以下功能：

- 算术运算(ADD, ADC, SUB, SBC, DAA)
- 逻辑运算(AND, OR, XOR, CPL)
- 移位运算(RL, RR, RLC, RRC)
- 递增和递减(INC, DEC)
- 分支判断(SZ, SNZ, SIZ, SDZ...)

ALU 不仅可以储存数据运算的结果，还会改变状态寄存器的值。

状态寄存器 — STATUS

8 位的状态寄存器(0AH)，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)和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，而且还控制操作顺序。

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产生借位，则 C 被置位；反之，C 被清除。它也可被循环移位指令影响。
1	AC	如果在加法运算中低 4 位产生了进位或减法运算中低 4 位不产生借位，则 AC 被置位；反之，AC 被清除。
2	Z	如果算术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，则 OV 被置位；反之，OV 被清除
4	PDF	系统上电或执行“CLR WDT”指令，PDF 被清除；执行“HALT”指令，PDF 被置位。
5	TO	系统上电、执行“CLR WDT”或“HALT”指令，TO 被清除；WDT 定时溢出，TO 被置位。
6, 7	—	未用，读数为“0”

STATUS (0AH) 寄存器

除了 PDF 和 TO 标志外，状态寄存器的其它位都可以用指令改变。任何对状态寄存器的写操作都不会改变 PDF 和 TO 的值。对状态寄存器的操作可能会导致与预期不一样的结果。TO 标志只受系统上电、看门狗溢出、“CLR WDT”指令或“HALT”指令的影响。PDF 标志只受系统上电、“CLR WDT”指令或“HALT”指令的影响。标志位 Z、OV、AC 和 C 反映的是最近一次操作的状态。

在进入中断程序或子程序调用时，状态寄存器不会被自动压入堆栈。如果状态寄存器的内容是重要的，而且子程序会影响状态寄存器的内容，那么程序员必须事先将 STATUS 的值保存好。

中断

HT46xU66 提供两个外部中断、两个内部定时/计数器中断、一个 UART 中断和一个多功能中断。多功能中断包含一个定时/计数器 2 中断、RTC 中断和时基中断。中断控制寄存器 0(INTC0: 0BH)、中断控制寄存器 1(INTC1: 1EH)和多功能中断控制寄存器 (MFIC: 2FH) 包含了中断控制位和中断请求标志，中断控制位用来设置中断允许/禁止。

只要有中断子程序被服务，其余的中断全部都被自动禁止(通过清除 EMI 位)，这种做法的目的在于防止中断嵌套。这时如果有其它中断发生，只有中断请求标志会被记录下来。如果在中断服务程序中有另一个中断需要响应，程序员可以置位 EMI、INTC0、INTC1 和 MFIC 所对应的位，以便进行中断嵌套。如果堆栈已满，则中断并不会被响应，一直到堆栈指针(SP)发生递减后才会响应。如果需要中断立即得到响应，应避免堆栈饱和。

位	符号	功 能
0	EMI	总中断控制位(1=允许; 0=禁止)
1	EEI0	外部中断 0 控制位(1=允许; 0=禁止)
2	EEI1	外部中断 1 控制位(1=允许; 0=禁止)
3	ETOI	定时/计数器 0 中断控制位(1=允许; 0=禁止)
4	EIF0	外部中断 0 请求标志(1=有; 0=无)
5	EIF1	外部中断 1 请求标志(1=有; 0=无)
6	T0F	定时/计数器 0 中断请求标志(1=有; 0=无)
7	—	只作内部测试用 使用时必须写入‘0’; 否则会发生不可预知的错误

INTC0 (0BH) 寄存器

位	符号	功 能
0	ETI1	定时/计数器 1 中断控制位(1=允许; 0=禁止)
1	EURI	UART 中断控制位(1=允许; 0=禁止)
2	EMFI	多功能中断控制位(1=允许; 0=禁止)
3, 7	—	未用, 读出为“0”
4	T1F	定时/计数器 1 中断请求标志(1=有; 0=无)
5	URIF	UART 中断请求标志(1=有; 0=无)
6	MFF	多功能中断请求标志(1=有; 0=无)

INTC1 (1EH) 寄存器

位	符号	功 能
0	ET2I	定时/计数器 2 中断控制位(1=允许; 0=禁止)
1	ETBI	时基中断控制位(1=允许; 0=禁止)
2	ERTI	RTC 中断控制位(1=允许; 0=禁止)
3, 7	—	未用, 读出为“0”
4	T2F	定时/计数器 2 中断请求标志(1=有; 0=无)
5	TBF	时基中断请求标志(1=有; 0=无)
6	RTF	RTC 中断请求标志(1=有; 0=无)

MFIC (2FH) 寄存器

所有的中断都具有唤醒能力。当有中断被服务，系统会将程序计数器的内容压入堆栈，然后再跳转至中断服务程序的入口。但这时只有程序计数器的内容被压入堆栈，如果其它寄存器和状态寄存器的内容会被中断程序改变，从而会破坏主程序的控制流程的话，程序员应该事先将这些数据保存起来。

外部中断是由 $\overline{INT0}/\overline{INT1}$ 引脚电平变化触发的(可由配置设置为上升沿触发、下降沿触发或两者皆可触发)，其中断请求标志位(EIF0(EIF1; INTC0 的第 4、5 位)会被置位。如果中断允许，且堆栈未满，当发生外部中断时，会产生地址 004H/008H 的子程序调用；而中断请求标志 EIF0(EIF1 和总中断控制位 EMI 会被清除，以禁止其它中断响应。

内部定时/计数器 0 中断是由定时/计数器 0 溢出触发的，其中断请求标志(T0F; INTC0 的第 6 位)会被置位。如果中断允许，且堆栈未满，当发生定时/计数器 0 中断时，会产生地址 00CH 的子程序调用；而中断请求标志 T0F 和总中断控制位 EMI 会被清除，以禁止其它中断响应。

内部定时/计数器 1 和定时/计数器 2 的运作方式与之相同，定时/计数器 1 的中断请求标志位是 T1F(INTC1 的第 4 位)，而它的子程序调用的地址是 10H 单元。定时/计数器 2 的中断请求标志位是 MFF (INTC1 的第 6 位) 和 T2F (MFIC 的第 4 位)，它的子程序调用的地址是 18H 单元。中断请求标志 T1F、MFF 和总中断控制位 EMI 会被清除，以禁止其它中断响应。中断请求标志 T2F 不会自动清除，必须由程序员用软件清除。

异步串行中断是由发送数据寄存器标志位 TXIF、接收数据寄存器标志位 RXIF、发送标志位寄存器 TIDF 或接收标志寄存器 RIDF 置位触发的，其中断请求标志 (URF; INTC1 的第 5 位) 会被置位。如果中断允许，且堆栈未满，当 TXIF、RXIF、TIDF 或 RIDF 被置位，会产生地址 014H 的子程序调用；而中断请求标志 URIF 和总中断控制位 EMI 会被清除，以禁止其它中断响应。

多功能中断是由定时/计数器 2 溢出 (T2F; MFIC 的第 4 位)、RTC 溢出 (RTF; MFIC 的第 6 位) 或时基溢出 (TBF; MFIC 的第 5 位) 触发的，其中断请求标志位 (MFF; INTC1 的第 6 位) 会被置位。如果中断允许，且堆栈未满，当 MFF 被置位，会产生地址 018H 的子程序调用；而中断请求标志 MFF 和总中断控制位 EMI 会被清除，以禁止其它中断响应。T2F、TBF 和 RTF 表明发生了定时/计数器 2、时基或 RTC 中断，中断标志 T2F、TBF 和 RTF 不会自动清除，必须由程序员用软件清除。

在执行中断子程序期间，其它的中断请求会被屏蔽，直到执行 RETI 指令或 EMI 和相关中断控制位被置位(当然，此时堆栈未满)。如果要从中断子程序返回，只要执行 RET 或 RETI 指令即可。其中，RETI 指令会自动置位 EMI，以允许中断服务，而 RET 则不会。

如果中断在两个连续的 T2 脉冲的上升沿之间发生，且中断响应允许，那么在下两个 T2 脉冲之间，该中断会被服务。如果同时发生中断请求，其优先级如下表，这些中断都可以通过清除 EMI 位来进行屏蔽。

中断源	优先级	中断向量
外部中断 0	1	004H
外部中断 1	2	008H
定时/计数器 0 中断	3	00CH
定时/计数器 1 中断	4	010H
UART 中断	5	014H
多功能中断 (定时/计数器 2、RTC 或时基溢出)	6	018H

中断控制寄存器 INTC0 (0BH)，由定时/计数器 0 中断请求标志 (T0F)、外部中断 1 请求标志 (EIF1)、外部中断 0 请求标志(EIF0)、定时/计数器 0 中断允许位(ET0I)、外部中断 1 允许位(EEI1)、外部中断 0 允许位 (EEI0) 和总中断控制位 EMI 组成。

中断控制寄存器 INTC1 (1EH)，由多功能中断请求标志 (MFF)、UART 中断请求标志 (URIF)、定时/计数器 1 中断请求标志 (T1F)、多功能中断允许位 (EMFI)、UART 中断允许位 (EURI) 和定时/计数器 1 中断允许位 (ETI1) 组成。

多功能中断控制寄存器 MFIC (2FH)，时基中断请求标志 (TBF)、RTC 中断请求标志 (RTF)、定时/计数器 2 中断请求标志 (T2F)、时基中断允许位 (ETBI)、RTC 中断允许位 (ERTI) 和定时/计数器 2 中断允许位 (ET2I) 位组成。

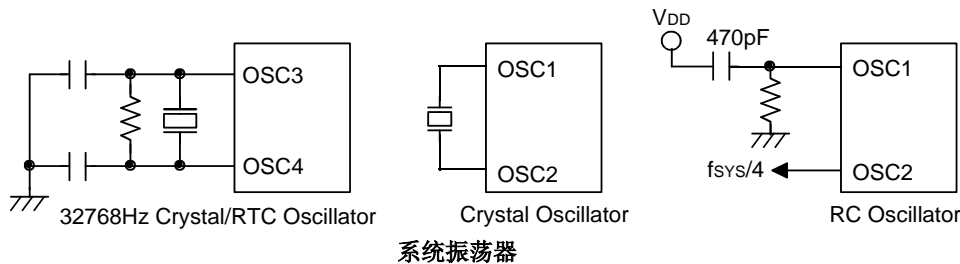
EMI、EEI0、EEI1、EEI2、ET0I、ET1I 和 EMFI 用来控制中断的允许/禁止状态。这些控制位可以用来屏蔽正在进行中断服务程序时发生的其它中断请求。一旦中断请求标志(EIF0、EIF1、T0F、T1F、URIF、MFF)被置位，会一直保留在 INTC0 和 INTC1 寄存器中，直到中断被响应或用软件指令清除为止。

定时/计数器 2 溢出中断标志 (T2F; MFIC 的第 4 位)、RTC 溢出中断标志 (RTF; MFIC 的第 6 位) 和时基溢出中断标志 (TBF; MFIC 的第 5 位) 用来表明发生的相应的中断。这些位不会自动清除，必须由程序员用软件清除。此外 MFIC (2FH) 寄存器还包括定时/计数器 2 中断允许位 (ET2I)、时基中断允许位 (ETBI) 和 RTC 中断允许位 (ERTI)。

建议不要在中断服务程序中使用“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。如果只剩下一层堆栈，而中断不能被很好地控制，原先的控制序列很可能因为在中断子程序中执行“CALL”指令而使堆栈溢出，从而发生混乱。

振荡电路

HT46xU66 有三种振荡方式可以做为系统时钟：外部 RC 振荡、外部晶体振荡和外部 32768Hz 晶体振荡，可以通过配置选项设定。HALT 模式会停止系统振荡器(当选择外部 RC 振荡或外部晶体振荡时)，并忽视任何外部信号以降低功耗。但是 32768Hz 的晶体振荡在 HALT 模式下仍会继续作用。如果选择 32768Hz 振荡做为系统振荡，在 HALT 模式下系统振荡不会停止，但是指令会停止运行。32768Hz 的晶体振荡还可以做为内部计数器的时钟源，即使进入 HALT 模式，这些计数器(RTC、时基、WDT)还会继续作用。



注：32.768kHz 可用作 WDT 时钟或内部时钟
 当使用 32.768kHz 的晶振时，外部的电阻电容不是必须的。但在精确的 RTC 应用场合，电阻电容可以用来消除晶振制造带来的误差。

如果选用外部 RC 振荡方式，在 OSC1 与 VSS 之间需要接一个外部电阻，其阻值为 24kΩ 到 1MΩ；而 OSC2 上会输出系统频率的 4 分频信号，可用于同步外部逻辑。RC 振荡方式是一种低成本方案，但是，RC 振荡频率会随着 VDD、温度和芯片自身参数的漂移而产生误差。因此，在需要精确振荡频率做为计时操作的场合，并不适合使用 RC 振荡方式。

如果选用晶体振荡方式，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的反馈和相移，除此之外，不再需要其它外部元件。另外，在 OSC1 和 OSC2 之间也可使用谐振器来取代晶体振荡器，但是在 OSC1 和 OSC2 需要多连接两个电容。

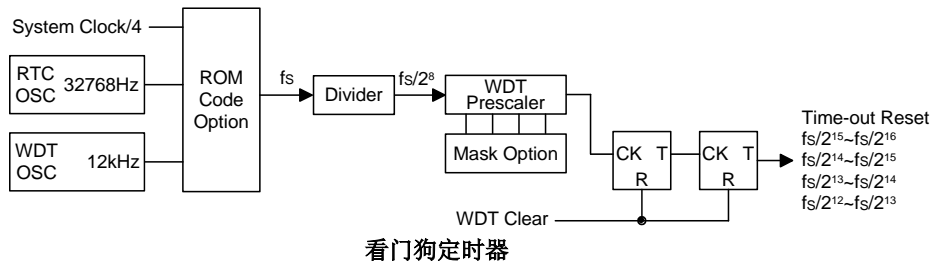
另外一个专为实时时钟设计的振荡电路。如果使用 RTC 振荡，那么只要在 OSC3 与 OSC4 之间接一个 32.768kHz 的晶体，不需要其它外部器件。

RTC 振荡器可以通过“QOSC”(RTCC 的第 4 位)设置快速起振。建议在系统上电时开启快速振荡，并在 2 秒钟后关闭。

WDT 振荡是一个独立的内置振荡电路，不需要外接器件。当系统进入省电模式，系统振荡会停止，但 WDT 振荡会继续作用，其振荡周期一般为 65μs@5V。WDT 振荡器可以由配置选项设置为打开或关闭。

看门狗定时器

WDT 的时钟来源可由配置选项设置为内部 RC 振荡(WDT 振荡器)、指令时钟(系统时钟 4 分频)或 RTC 振荡。WDT 主要用来防止程序运行故障和程序跳入一死循环而导致不可预测的结果。WDT 可由配置选项设置为打开或关闭,如果在关闭状态,所有与 WDT 有关的指令操作都是没有作用的。



如果 WDT 时钟源为内部 WDT 振荡(RC 振荡周期一般为 $65\mu\text{s}@5\text{V}$),该频率可再经过 $2^{12}\sim 2^{15}$ (由配置选项设置)的分频。最小的 WDT 溢出周期大约是 $300\text{ms}\sim 600\text{ms}$ 。但溢出时间会因为温度、VDD 以及芯片参数的变化而变化。如果再用 WDT 预分频器,则可以得到更长的溢出周期。如果 WDT 的溢出时间选为 2^{15} 分频,最大的溢出时间可达到 $2.1\text{s}\sim 4.3\text{s}$ (分频系数为 $2^{15}\sim 2^{16}$)。

如果 WDT 的时钟源为指令时钟,则在 HALT 状态时,WDT 会停止计数而失去保护功能;此时只能靠外部逻辑复位来重新启动系统。如果系统运用在强干扰的环境中,建议选用内部 WDT 振荡器,因为 HALT 模式会使系统时钟停止,看门狗也就失去了保护的功能。

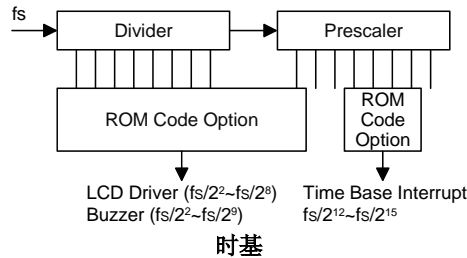
在正常运行时,WDT 溢出会使系统复位并置位 TO 标志;但在 HALT 模式下,WDT 溢出只产生“热复位”,只有程序计数器 Program Counter 和堆栈指针 SP 被复位。要清除 WDT 的值可以有三种方法:外部复位(低电平输入到 $\overline{\text{RES}}$ 端)、清除看门狗指令或 HALT 指令。清除看门狗指令有“CLR WDT”和“CLR WDT1”、“CLR WDT2”二组指令。这两组指令中,只能选择其中一组,由配置选项决定。如果选择“CLR WDT”,那么只要执行“CLR WDT”指令就会清除 WDT。如果选择“CLR WDT1”和“CLR WDT2”,那么二条指令要交替使用才会清除 WDT,否则,WDT 会由于溢出而使系统复位。

多功能定时器

系统为 WDT、时基和 RTC 提供了具有不同溢出周期的多功能定时器。多功能定时器由一个 8 级分频器和一个 7 位预分频器组成。其时钟源可以是 WDT OSC、RTC OSC 或指令时钟(系统时钟四分频)。多功能定时器还为 LCD 驱动电路提供可选择的频率信号(范围从 $f_s/2^2\sim f_s/2^8$),并为蜂鸣器输出电路提供可选择的频率信号(范围从 $f_s/2^2\sim f_s/2^9$),频率由配置选项决定。为了正确地显示,建议选择 4kHz 左右的信号作为 LCD 驱动信号。

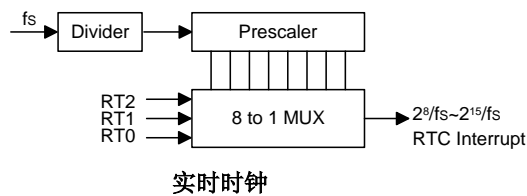
时基

时基指的是用一个周期性的溢出来产生一个有规律的内部中断。溢出周期的范围为 $2^{12}/fs \sim 2^{15}/fs$ ，由配置选项确定。当时基发生溢出，其中断请求标志(TBF; MFIC 的第 5 位、MFF; INTC1 的第 6 位)会被置位，如果中断允许，且堆栈未满，那么就会产生一个地址 018H 的子程序调用。时基溢出信号可作为定时/计数器 1 时钟源，以便得出更长的时间溢出周期。



实时时钟

实时时钟(RTC)的工作情况和时基一样。它是用来提供一个有规律的内部中断。它的溢出周期范围为 $fs/2^8 \sim fs/2^{15}$ ，可通过软件编程实现。写数据到 RT2、RT1、RT0 (RTCC 的第 2、1、0 位; 09H) 将产生各种溢出周期。当 RTC 发生溢出，其中断请求标志(RTF; MFIC 的第 6 位、MFF; INTC1 的第 6 位)被置位，如果中断允许，且堆栈未满，那么就会产生一个地址 018H 的子程序调用。RTC 溢出可作为定时/计数器 0 时钟源，以便得出更长的时间溢出周期。



RT2	RT1	RT0	RTC 实时时钟分频级数
0	0	0	2^8^*
0	0	1	2^9^*
0	1	0	2^{10}^*
0	1	1	2^{11}^*
1	0	0	2^{12}
1	0	1	2^{13}
1	1	0	2^{14}
1	1	1	2^{15}

注：“*”不建议使用

暂停模式

暂停模式是由 HALT 指令来实现的，暂停模式时系统状态如下：

- 系统振荡器停振，但 WDT 振荡器会继续振荡(如果选择 WDT 振荡或 RTC 振荡)。
- RAM 和寄存器内容保持不变。
- WDT 被清除并重新开始计数(如果 WDT 时钟来源选择 WDT 振荡或 RTC 振荡)。
- 所有输入/输出口都保持其原有状态。
- 置位 PDF 标志，清除 TO 标志。
- LCD 驱动器仍然运行（如果选择 WDT OSC 或 RTC OSC）。

以下操作可以使系统离开暂停模式：外部复位、中断、PA 口下降沿信号或看门狗定时器溢出。其中，外部复位会使系统初始化，WDT 溢出则会发生“热复位”。通过检测 TO 和 PDF 标志，即可了解系统复位的原因。PDF 标志可由系统上电或执行“CLR WDT”指令清除，由 HALT 指令置位。TO 标志由 WDT 溢出置位，同时产生唤醒，但只有程序计数器 Program Counter 和堆栈指针 SP 被复位，其它都保持其原有的状态。

PA 口唤醒和中断唤醒可作为正常运行的继续。PA 口的每一位都可以由配置选项设置为唤醒功能。如果是由输入/输出口唤醒，程序会从下一条指令开始运行。如果是由中断唤醒，可能会发生两种情况：如果中断禁止或中断允许但堆栈已满，程序将会从下一条指令开始运行；如果中断允许且堆栈未满，则会产生一般的中断响应。如果在进入 HALT 模式之前，中断请求标志位已被置“1”，则中断唤醒功能被禁止。

当发生唤醒，系统需要额外花费 $1024t_{SYS}$ (系统时钟周期)的时间，才能重新正常运行，也就是说，唤醒之后会插入一个等待周期。如果唤醒是由中断产生的话，则实际中断子程序的执行会延迟一个以上的周期。如果唤醒导致下一条指令执行，那么在等待周期执行完成之后，会立即执行该指令。

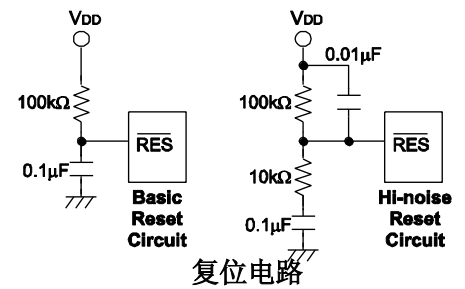
为减小功耗，在进入暂停模式之前，应小心处理所有的输入/输出口状态。

复位

总共有三种方法会产生初始复位:

- 正常运行时由 $\overline{\text{RES}}$ 引脚发生复位。
- 在暂停模式由 $\overline{\text{RES}}$ 引脚发生复位。
- 正常运行时由看门狗定时器溢出发生复位。

暂停模式中的看门狗定时器溢出与其它系统复位状况不同, 因为看门狗定时器溢出会执行“热复位”, 只有程序计数器 Program Counter 和堆栈指针 SP 被复位, 而系统其它部分都保持原有状态。在其它复位状态下, 某些寄存器不会改变。在初始复位时, 大部分寄存器会复位成初始的状态。通过检测 PDF 和 TO 标志, 即可判断出各种不同的复位原因。



注: 在一般应用场合可使用基本复位电路, 而在强干扰环境中, 建议使用抗杂讯复位电路。

TO	PDF	复位原因
0	0	上电时 $\overline{\text{RES}}$ 发生复位
u	u	正常运行时 $\overline{\text{RES}}$ 发生复位
0	1	暂停模式下 $\overline{\text{RES}}$ 发生复位
1	u	正常运行时 WDT 溢出
1	1	暂停模式下 WDT 溢出

注: “u”表示不变

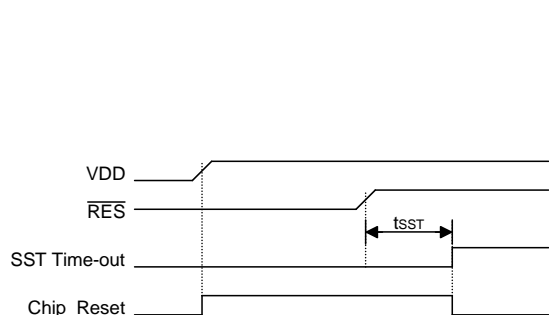
为了保证系统振荡器起振并稳定运行, 系统复位(包括上电复位、WDT 溢出或由 $\overline{\text{RES}}$ 端复位)或由暂停状态唤醒时, 系统启动定时器(SST)提供了一个额外的延迟时间, 共 1024 个系统时钟周期。

系统复位时, SST 会被加在复位延时中; 由暂停模式唤醒也会加入 SST 延迟。

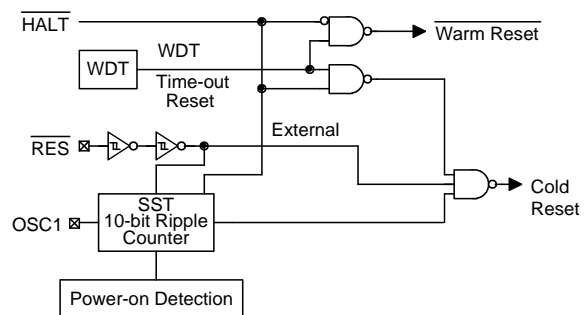
系统复位(包括上电复位、正常运行时 WDT 溢出或由 $\overline{\text{RES}}$ 端复位)需要额外增加一个加载配置选项(Option)的时间。

系统复位时各功能单元的状态如下所示:

程序计数器	000H
中断	禁止
预分频器、分频器	清除
WDT、RTC、时基	清除, 在主系统复位后, WDT 开始计数
定时/计数器	停止
输入/输出口	输入模式
堆栈指针 SP	指向堆栈顶部



复位时序图



复位配置

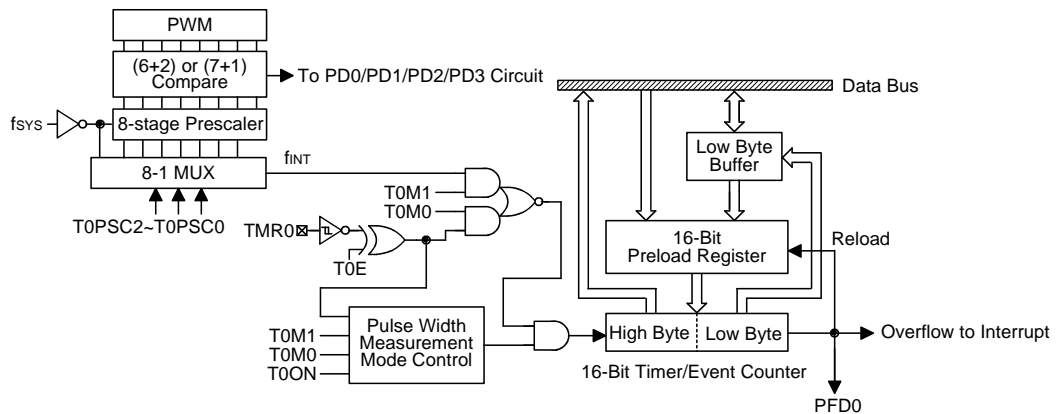
有关寄存器的状态如下:

寄存器	复位 (上电复位)	WDT 溢出 (正常运行)	RES端复位 (正常运行)	RES端复位 (暂停模式)	WDT 溢出 (暂停模式)*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	0000 0000	0000 0000	0000 0000	0000 0000	00u0 00uu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
Program Counter	0000H	0000H	0000H	0000H	0000H
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
RTCC	--00 0111	--00 0111	--00 0111	--00 0111	--uu uuuu
STATUS	--00 xxxx	--lu uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TMR0H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0L	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	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
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
PWM0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM1	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM2	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PWM3	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
TBHP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ADRL	xxxx ---	xxxx ---	xxxx ---	xxxx ---	uuuu ---
ADRH	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR	0100 0000	0100 0000	0100 0000	0100 0000	uuuu uuuu
ACSR	1--- --00	1--- --00	1--- --00	---- --00	u--- --uu
TMR2	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR2C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
MFIC	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
USR	0000 1011	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TXR/RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

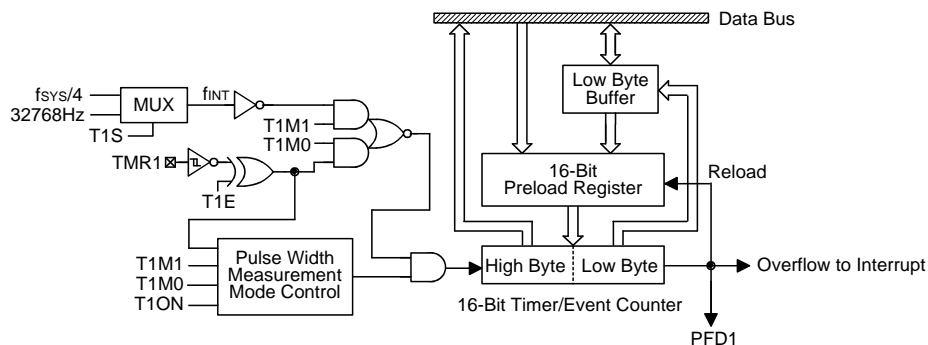
注: 1.“*”表示“热复位”; 2.“u”表示不变化; 3.“x”表示不确定。

定时/计数器

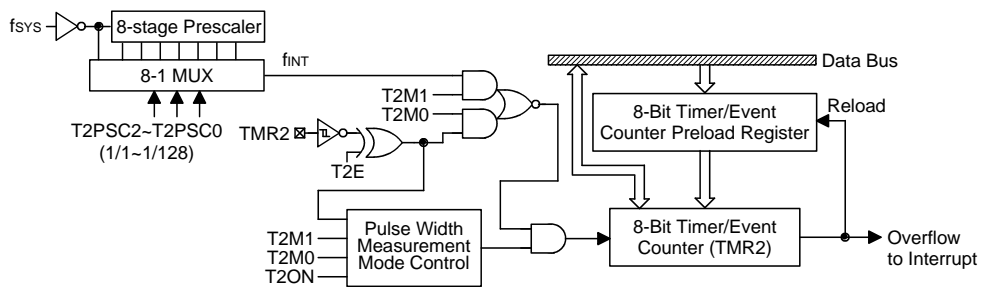
HT46xU66 系统提供三个定时/计数器(TMR0、TMR1、TMR2)。TMR0 是一个 16 位可编程的向上计数的计数器，其时钟来源可以是外部信号输入或内部时钟(f_{SYS})。TMR1 是一个 16 位可编程的向上计数的计数器，其时钟来源可以是外部信号输入或内部时钟($f_{SYS}/4$ 或 32768Hz 振荡，由配置选项设置)。TMR2 是一个 8 位可编程的向上计数的计数器，其时钟来源可以是外部信号输入或内部时钟(f_{SYS})。外部信号输入可以用来计数外部事件、测量时间间隔、测量脉冲宽度或产生一个精确的时基信号。



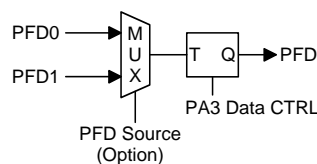
定时/计数器 0



定时/计数器 1



定时/计数器 2



PFD 来源选择

总共有八个与定时/计数器 0/1/2 有关的寄存器，TMR0H(0CH)、TMR0L(0DH)、TMR0C(0EH)、TMR1H(0FH)、TMR1L(10H)、TMR1C(11H)、TMR2(2CH)和 TMR2C(2DH)。写入 TMR0L(TMR1L)只能将数据写到低字节缓冲器，而写入 TMR0H(TMR1H)会把指定数据和低字节缓冲器的数据写到 TMR1H 和 TMR1L 中。

定时/计数器 0/1 预置寄存器的内容只有在写入 TMR0H(TMR1H)时才会被改变而写 TMR0L(TMR1L)不会改变预置寄存器的值。读取 TMR0H(TMR1H)会把 TMR0H(TMR1H)的内容送至目标单元，而 TMR0L(TMR1L)的值被送至低字节缓冲器中。读 TMR0L(TMR1L)将读取低字节缓冲器的值。写 TMR2 会将初始值装入到定时/计数器 2 的预置寄存器中，而读 TMR2 则会取得定时/计数器 2 的内容。TMR0C(TMR1C、TMR2C)是定时/计数器 0 (1、2) 控制寄存器，用来定义定时/计数器一些选项。

T0M0、T0M1(TMR0C)，T1M0、T1M1(TMR1C)和 T2M0、T2M1 (TMR2C) 用来定义定时/计数器的工作模式。外部事件计数模式是用来记录外部事件的，其时钟来源为外部 TMR0/TMR1/TMR2 引脚输入。定时器模式是一个常用模式，其时钟来源为内部时钟。脉宽测量模式可以测量 TMR0/TMR1/TMR2 引脚高/低电平的脉冲宽度，其时钟来源为内部时钟。

无论是定时模式还是外部事件计数模式，一旦开始计数，定时/计数器 0/1 会从寄存器当前值向上计到 0FFFFH；定时/计数器 2 会从寄存器当前值向上计到 0FFH。一旦发生溢出，定时/计数器会从预置寄存器中重新加载初值，并开始计数；同时置位中断请求标志(T0F, INTC0 的第 6 位；T1F, INTC1 的第 4 位；T2F, MFIC 的第 4 位；MFF, INTC1 的第 6 位)。

在脉宽测量模式，当 T0ON/T1ON/T2ON 与 T0E/T1E/T2E 是 1 时，只要 TMR0/TMR1/TMR2 引脚有一个上升沿信号(如果 T0E/T1E/T2E 是 0，则为下降沿信号)，定时/计数器就会开始计数，直到 TMR0/TMR1/TMR2 脚电平恢复，同时 T0ON/T1ON/T2ON 被清零。测量的结果会保存在寄存器中，直到有新的测量开始。换句话说，一次只能测量一个脉冲宽度。重新置位 T0ON/T1ON/T2ON 后，可以继续测量。注意，在该模式下，定时/计数器是跳变触发而不是电平触发。当计数器溢出时，定时/计数器会从预置寄存器中重新加载初值，并置位中断请求标志，这与其它两种模式一样。

要启动计数器，只要置位 T0ON/T1ON /T2ON(TMR0C/TMR1C/TMR2C 的第 4 位)。在脉宽测量模式下，T0ON/T1ON/T2ON 在测量结束后会被自动清除；但在另外两种模式中，T0ON/T1ON/T2ON 只能由指令来清除。定时/计数器 0/1/2 的溢出可以作为唤醒信号，定时/计数器 0/1 可以提供给 PFD (可编程分频输出)使用。如果 PA3 选择为 PFD 输出，有两种模式选择：一种是选择 PFD0 做为 PFD 输出，另一种是选择 PFD1 做为 PFD 输出，PFD0、PFD1 分别是定时/计数器 0、定时/计数器 1 的溢出信号。不管是什么模式，只要写 0 到 ET0I、ET1I 或 ET2I 即可禁止定时/计数器中断服务。当使用 PFD 功能时，执行“SET [PA].3”可以打开 PFD 输出，执行“CLR [PA].3”则关闭 PFD 输出。

在定时/计数器停止计数时，写数据到定时/计数器的预置寄存器中，同时会将该数据写入到定时/计数器。但如果在定时/计数器运行时这么做，数据只能写入到预置寄存器中，直到发生溢出时才会将数据从预置寄存器加载到定时/计数器寄存器。读取定时/计数器时，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。

由于系统上电时 TMR0/TMR1/TMR2 寄存器处于未知状态，在打开定时/计数器进行相关操作时，建议将数据先写入 TMR0/TMR1/TMR2 寄存器中。

TMR0C/TMR2C 的第 0~2 位用来定义内部时钟预分频级数，定义如上表所示。定时/计数器的溢出信号可做为 PFD 输出。

位	符号	功能
0 1 2	T0PSC0 T0PSC1 T0PSC2	定义预分频器级数, T0PSC2, T0PSC1, T0PSC0= 000: $f_{INT}=f_{SYS}$ 001: $f_{INT}=f_{SYS}/2$ 010: $f_{INT}=f_{SYS}/4$ 011: $f_{INT}=f_{SYS}/8$ 100: $f_{INT}=f_{SYS}/16$ 101: $f_{INT}=f_{SYS}/32$ 110: $f_{INT}=f_{SYS}/64$ 111: $f_{INT}=f_{SYS}/128$
3	T0E	定义定时/计数器 TMR0 的触发方式 在事件计数模式 (T0M1, T0M0) = (0, 1): 1: 在下降沿计数 0: 在上升沿计数 在脉冲宽度测量模式 (T0M1, T0M0) = (1, 1): 1: 在上升沿开始计数, 下降沿停止计数 0: 在下降沿开始计数, 上升沿停止计数
4	T0ON	打开/关闭定时/计数器(0=关闭, 1=打开)
5	—	未用, 读数为“0”
6 7	T0M0 T0M1	定义工作模式(T0M1, T0M0): 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

TMR0C (0EH) 寄存器

位	符号	功能
0~2	—	未用, 读数为“0”
3	T1E	定义定时/计数器 TMR1 的触发方式 在事件计数模式 (T1M1, T1M0) = (0, 1): 1: 在下降沿计数 0: 在上升沿计数 在脉冲宽度测量模式 (T1M1, T1M0) = (1, 1): 1: 在上升沿开始计数, 下降沿停止计数 0: 在下降沿开始计数, 上升沿停止计数
4	T1ON	打开/关闭定时/计数器(1=打开, 0=关闭)
5	T1S	内部时钟来源选项 (0=系统时钟或是系统时钟 4 分频, 1=RTC 输出)
6 7	T1M0 T1M1	定义工作模式(T1M1, T1M0): 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

TMR1C (11H) 寄存器

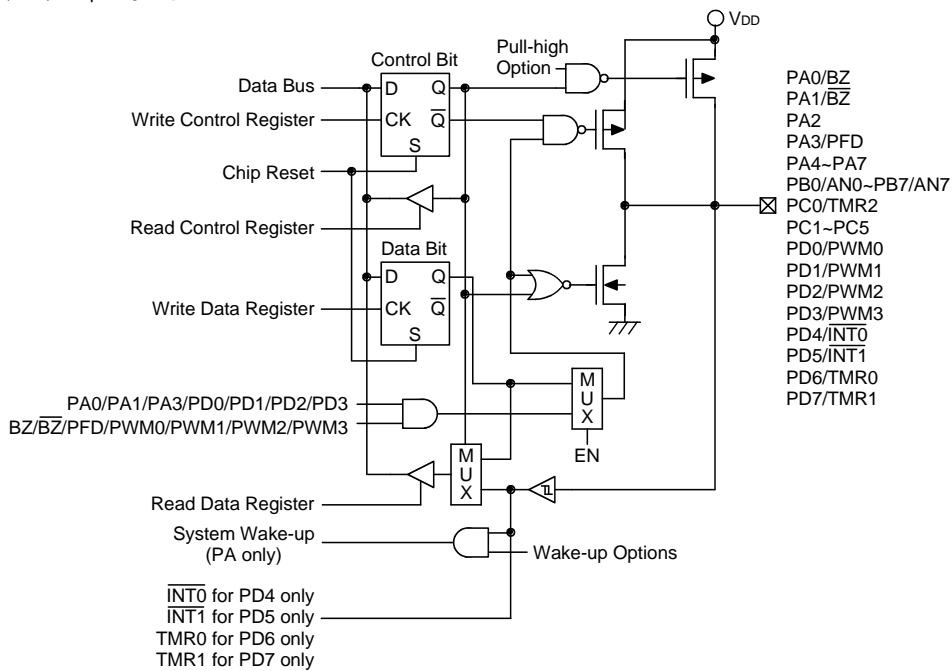
位	符号	功能
0 1 2	T2PSC 0 T2PSC 1 T2PSC 2	定义预分频器级数, T2PSC 2, T2PSC 1, T2PSC 0= 000: $f_{INT}=f_{SYS}$ 001: $f_{INT}=f_{SYS}/2$ 010: $f_{INT}=f_{SYS}/4$ 011: $f_{INT}=f_{SYS}/8$ 100: $f_{INT}=f_{SYS}/16$ 101: $f_{INT}=f_{SYS}/32$ 110: $f_{INT}=f_{SYS}/64$ 111: $f_{INT}=f_{SYS}/128$
3	T2E	定义定时/计数器 TMR2 的触发方式 在事件计数模式 (T2M1, T2M0) = (0, 1): 1: 在下降沿计数 0: 在上升沿计数 在脉冲宽度测量模式 (T2M1, T2M0) = (1, 1): 1: 在上升沿开始计数, 下降沿停止计数 0: 在下降沿开始计数, 上升沿停止计数
4	T2ON	打开/关闭定时/计数器(0=关闭, 1=打开)
5	—	未用, 读数为“0”
6 7	T2M0 T2M1	定义工作模式(T2M1, T2M0): 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

TMR2C (2EH) 寄存器

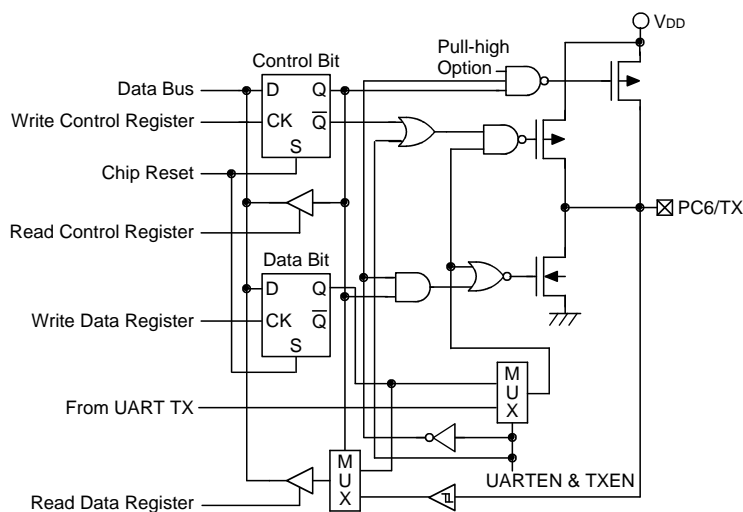
输入/输出口

HT46xU66 有 32 位双向输入/输出口，记为 PA、PB、PC 和 PD，其分别对应 RAM 地址[12H]，[14H]，[16H]和[18H]，所有端口都可以进行输入/输出操作。输入时，端口没有锁存功能，输入信号必须在 MOV A, [m](m=12H、14H、16H 或 18H)指令的 T2 上升沿到来前准备好；输出时，端口有锁存功能，端口上的数据会保持不变直到执行下一个写入操作。

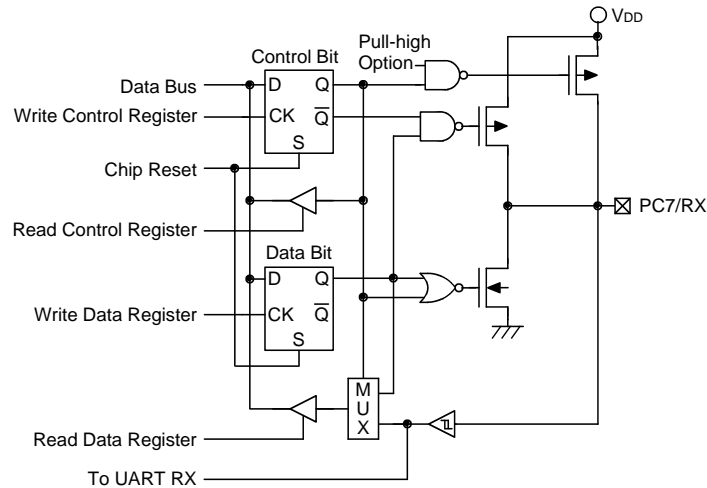
每个输入/输出口都有一个控制寄存器(PAC, PBC, PCC, PDC)，用来控制输入/输出状态。利用控制寄存器，可对 CMOS 输出、带或不带上拉电阻的斯密特触发输入通过软件动态地进行改变。做为输入时，对应的控制寄存器应设置为“1”。输入信号来源也取决于控制寄存器，如果控制寄存器的值为“1”，那么读取的是引脚状态；如控制寄存器的值为“0”，则读取的是内部锁存器的值。后者可能会在‘读-修改-写’指令中发生。做为输出时，只能采用 CMOS 输出。控制寄存器对应 RAM 地址 13H、15H、17H 和 19H。



输入/输出口



PC6/TX 输入/输出口



PC7/RX 输入/输出口

系统复位之后，这些输入/输出口会是高电平或浮空状态(由上拉电阻选项决定)。每一个输入/输出锁存位都能用“SET [m].i”或“CLR [m].i”指令置位或清除。

有些指令会先输入数据，然后进行输出操作。例如：“SET [m].i”，“CLR [m].i”，“CPL [m]”，“CPLA[m]”这些指令会先将整个端口状态读入 CPU 中，接着执行所定义的运算(位操作)，然后再将结果写入锁存器或累加器中。

PA 的每一个口都具有唤醒系统的能力。

所有的输入/输出口都有上拉电阻选项。一旦选择了上拉电阻选项，输入/输出口就加上了上拉电阻。如果不选择上拉电阻，必须注意在输入模式下，若输入/输出口会产生浮空状态。

PA0、PA1、PA3、PD4、PD5、PD6、PA7 分别与 BZ、 \overline{BZ} 、PFD、 $\overline{INT0}$ 、 $\overline{INT1}$ 、TMR0、TMR1 共用引脚。PC0、PC6 和 PC7 分别与 TMR2、TX 和 RX 共用引脚。

PA0、PA1 与 BZ、 \overline{BZ} 共用引脚，如果选择 BZ/ \overline{BZ} 功能，则 PA0/PA1 在输出模式时的输出信号将是由内部多功能定时器产生的蜂鸣器信号，而在输入模式始终保持其原来的功能。一旦选择 BZ/ \overline{BZ} 功能，蜂鸣器的输出信号只受 PA0 数据寄存器控制。PA0/PA1 的输入/输出功能如下所示：

PAC 寄存器 PAC0	PAC 寄存器 PAC1	PA 寄存器 PA0	PA 寄存器 PA1	输出
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	1	X	PA0=input, PA1=BZ
1	0	0	X	PA0=input, PA1=0
1	1	X	X	PA0=input, PA1=input

注：“X”任意值

“D”代表数据“0”或“1”

PA3 与 PFD 共用引脚，如果选择 PFD 功能，则 PA3 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，而在输入模式始终保持其原来的功能。一旦选择 PFD 功能，PFD 的输出信号只受 PA3 数据寄存器控制。向 PA3 数据寄存器写入“1”，则输出 PFD 信号；向 PA3 数据寄存器写入“0”，则 PA3 输出为“0”。PA3 的输入/输出功能如下所示：

I/O 模式	I/P (正常)	O/P (正常)	I/P (PFD)	O/P (PFD)
PA3	逻辑输入	逻辑输出	逻辑输入	PFD (定时/计数器开启)

注：PFD 的输出频率是定时/计数器溢出频率的 1/2

PB 口可以用做 A/D 转换输入，A/D 转换功能将在下面说明。还有四个与 PD0/PD1/PD2/PD3 共用引脚的 PWM 输出。如果选择 PWM 功能，则 PD0/PD1/PD2/PD3 口会有 PWM0/PWM1/PWM2/PWM3 信号输出（PD0/PD1/PD2/PD3 为输出模式）。PD0 的输入/输出功能如下所示：

I/O 模式	I/P (正常)	O/P (正常)	I/P (PWM)	O/P (PWM)
PD0	逻辑输入	逻辑输出	逻辑输入	PWM0
PD1				PWM1
PD2				PWM2
PD3				PWM3

建议用软件将未使用和没有外接的输入/输出口设置为输出模式，以防止这些端口在输入浮空时增加系统的功耗。

PFD 的控制信号和输出频率如下所示：

定时/计数器	定时/计数器预置值	PA3 数据寄存器	PA3 引脚状态	PFD 输出频率
关闭	X	0	0	X
关闭	X	1	U	X
开启	N	0	0	X
开启	N	1	PFD	$f_{TMR}/[2 \times (M-N)]$

注：“X”表示未定义
 “U”表示未知
 对于 PFD0 或 PFD1，“M”等于“65536”
 “N”定时/计数器初始值
 “ f_{TMR} ”定时/计数器输入频率

PWM

HT46xU66 有 3 个或 4 个 PWM 输出通道，取决于选用的封装。通过改变 PWM 寄存器值，使脉宽调制输出不同频率的占空比，可用于马达控制。

每个 PWM 通道都有一个寄存器相关，对于有 3 个通道的寄存器为 PWM0、PWM1 和 PWM2。对于有 4 个通道的寄存器为 PWM0、PWM1、PWM2 和 PWM3。8 位数据寄存器包含一个模式下的全部占空比信息。通过配置选项选择为 7+1 模式或 6+2 模式，每个 PWM 调制周期被分为 2 个或 4 个周期。当选择为 PWM 输出，只需选择相应的模式并将合适的数据写入 PWM 寄存器即可，其输出波形将由硬件自动完成。

PWM 计数器的时钟来源为系统时钟(f_{SYS})。

封装	通道	PWM 模式	输出引脚	PWM 寄存器
52/56-pin	3	6+2 或 7+1	PD0/PD1/PD2	PWM0/PWM1/PWM2
100-pin	4	6+2 或 7+1	PD0/PD1/PD2/PD3	PWM0/PWM1/PWM2/PWM3

PWM 功能表

将 PWM 调制周期分为 2 个或 4 个周期可以产生出更高的 PWM 频率，以适应更宽的应用范围。PWM 时钟来源于系统时钟 f_{SYS} ，PWM 的数据为 8 位，PWM 输出频率为 $f_{SYS}/256$ 。当选择 7+1 模式，PWM 调制周期为 $f_{SYS}/128$ ；当选择 6+2 模式，PWM 调制周期为 $f_{SYS}/64$ 。

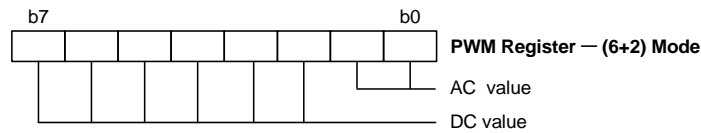
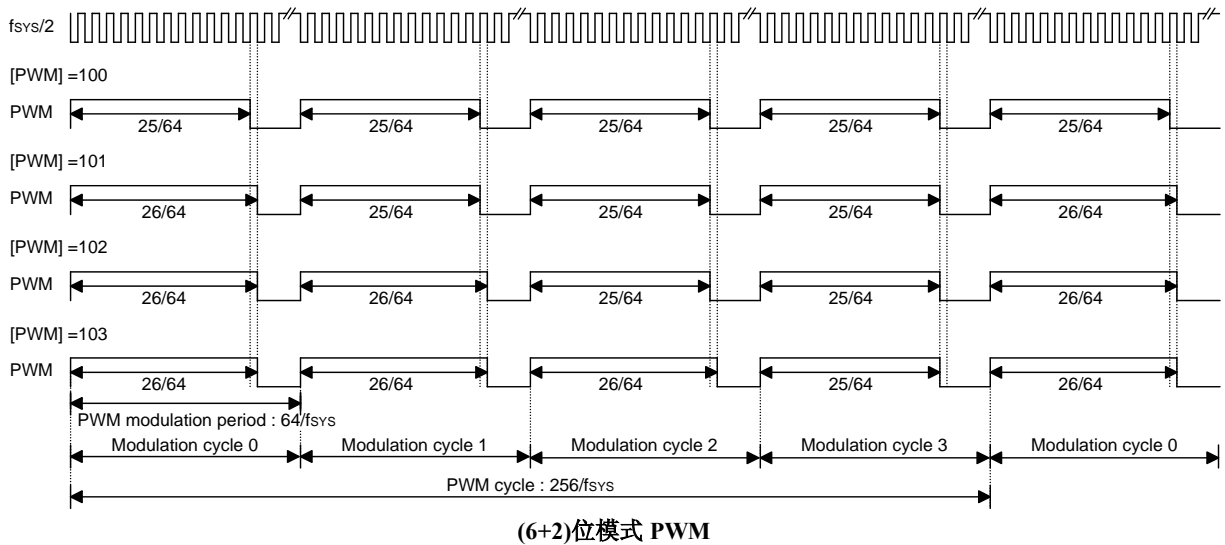
PWM 调制频率	PWM 周期频率	PWM 占空比
$f_{SYS}/64$ (6+2 模式)	$f_{SYS}/256$	[PWM]/256
$f_{SYS}/128$ (7+1 模式)		

● 6+2 PWM 模式

PWM 输出由 PWM 寄存器控制，完整的 PWM 周期包含 256 个时钟周期。在(6+2)位 PWM 模式中，一个 PWM 周期被分为四个调制周期(调制周期 0~调制周期 3)，每个调制周期有 64 个 PWM 输入时钟。在(6+2)位 PWM 模式中，PWM 寄存器被分为 2 个部分。第一部分是直流分量，由 PWM.7~PWM.2 控制；第二部分是交流分量，由 PWM.1~PWM.0 控制。在(6+2)位 PWM 模式中，每个调制周期的占空比见下表：

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

下图为 6+2 模式下 PWM 输出波形，它反映了一个 PWM 周期怎样分成 4 个调制周期，交流分量与 PWM 寄存器之间关系。



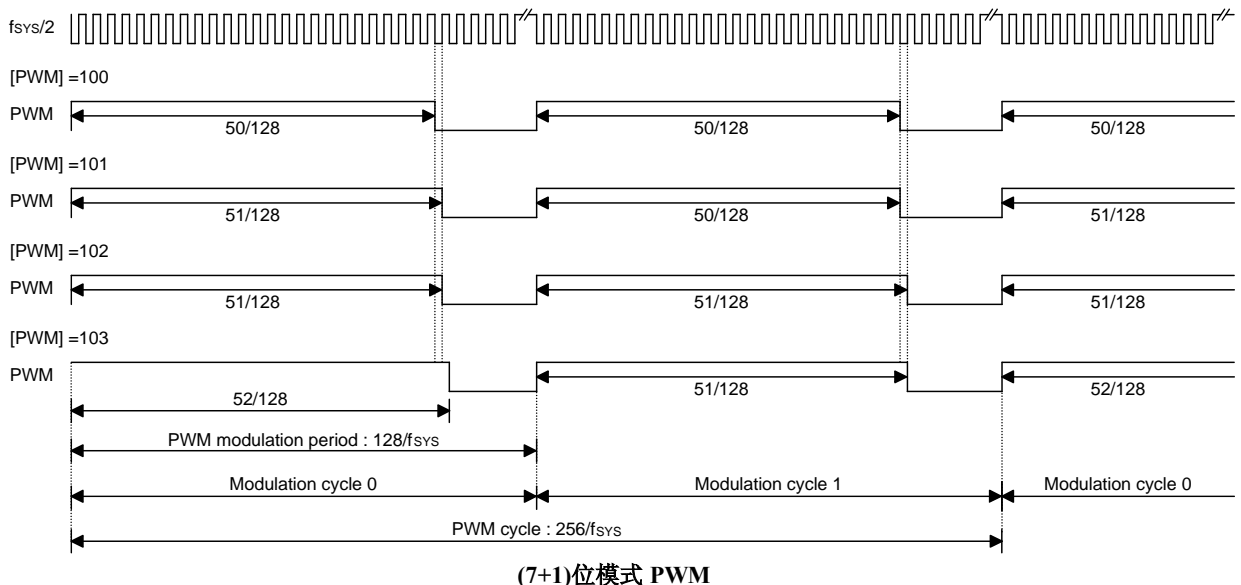
PWM (6+2 模式) 寄存器

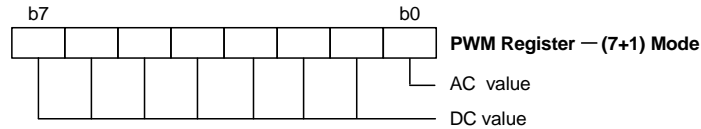
● 7+1 PWM 模式

PWM 输出由 PWM 寄存器控制，完整的 PWM 周期包含 256 个时钟周期。在(7+1)位 PWM 模式中，一个 PWM 周期被分为两个调制周期(调制周期 0~调制周期 1)，每个调制周期有 128 个 PWM 输入时钟。在(7+1)位 PWM 模式中，PWM 寄存器被分为 2 个部分。第一部分是直流分量，由 PWM.7~PWM.1 控制；第二部分是交流分量，由 PWM.0 控制。在(7+1)位 PWM 模式中，每个调制周期的占空比见下表：

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

下图为 7+1 模式下 PWM 输出波形，它反映了一个 PWM 周期怎样分成 2 个调制周期，交流分量与 PWM 寄存器之间关系。





PWM (7+1 模式) 寄存器

● PWM 输出控制

PWM 是与 PD 口共用引脚。如果选择 PWM 功能（由配置选项决定），则对应的 PWM 口会有 PWM 信号输出（相应的口为输出模式）。向 PD 数据寄存器写入“1”，则输出 PWM 信号；向 PD 数据寄存器写入“0”，则输出为低电平。PD 数据寄存器可作为 PWM 功能的开/关。注意：若在配置选项中选择 PWM 功能，但相应的 PD 控制寄存器写入“1”，则它们仍作为普通输入口使用。

下面示例程序表明怎样设置 PWM 输出，配置选项中已选择 PWM 输出。

```

clr  PDC.0      ; 设置 PD0 为输出
clr  PDC.1      ; 设置 PD1 为输出
clr  PDC.2      ; 设置 PD2 为输出
clr  PDC.3      ; 设置 PD3 为输出

set  pd.0       ; PD.0=1, 允许 PD0/PWM0 输出 PWM 波形
mov  a,64h      ; PWM0=100D=64H
mov  pwm0,a

set  pd.1       ; PD.1=1, 允许 PD1/PWM1 输出 PWM 波形
mov  a,65h      ; PWM1=101D=65H
mov  pwm1,a

set  pd.2       ; PD.2=1, 允许 PD2/PWM2 输出 PWM 波形
mov  a,66h      ; PWM2=102D=66H
mov  pwm2,a

set  pd.3       ; PD.3=1, 允许 PD3/PWM3 输出 PWM 波形
mov  a,67h      ; PWM3=103D=67H
mov  pwm3,a

clr  pd.0       ; 禁止 PWM0 输出 — PD.0 输出为低
clr  pd.1       ; 禁止 PWM1 输出 — PD.1 输出为低
clr  pd.2       ; 禁止 PWM2 输出 — PD.2 输出为低
clr  pd.3       ; 禁止 PWM3 输出 — PD.3 输出为低
    
```

A/D 转换

HT46xU66 有 8 个通道、12 位精度的 A/D 转换器，其参考电压为 VDD。与 A/D 转换有关的寄存器有 4 个，ADRL(24H)、ADRH(25H)、ADCR(26H)和 ACSR(27H)。ADRH 和 ADRL 是 A/D 转换结果的高字节和低字节寄存器，是只读寄存器。当完成 A/D 转换后，可从 ADRH 和 ADRL 读取 A/D 转换结果。ADCR 是 A/D 转换控制寄存器，用来定义 A/D 通道数量、模拟输入通道选择、A/D 转换开始控制和完成标志。如果要进行 A/D 转换，要先定义好 PB 口的设置，选择转换的模拟通道，然后给 START 控制位一个上升沿信号和一个下降沿信号(0→1→0)。完成 A/D 转换后， $\overline{\text{EOC}}$ 位会被清除。ACSR 是 A/D 时钟控制寄存器，用来选择 A/D 的时钟来源。

位	符号	功能
0 1	ADCS0 ADCS1	ADCS1, ADCS0: 选择 A/D 转换时钟源 00=系统时钟/2 01=系统时钟/8 10=系统时钟/32 11=未定义
2~6	—	未用，读出为“0”
7	TEST	只做为内部测试用

ACSR (27H) 寄存器

位	符号	功能
0 1 2	ACS0 ACS1 ACS2	ACS2, ACS1, ACS0 定义模拟输入通道 0, 0, 0: AN0 0, 0, 1: AN1 0, 1, 0: AN2 0, 1, 1: AN3 1, 0, 0: AN4 1, 0, 1: AN5 1, 1, 0: AN6 1, 1, 1: AN7
3 4 5	PCR0 PCR1 PCR2	定义 PB 口的设置 如果 PCR0、PCR1 和 PCR2 都为 0，则 A/D 转换电路被关闭以减小功耗
6	$\overline{\text{EOC}}$	A/D 转换结束标志(0: A/D 转换结束) 每次 BIT3-5 状态的改变都必须通过 START 信号来初始化 A/D 转换器，否则 $\overline{\text{EOC}}$ 可能会处于不确定状态，具体可参照“A/D 转换初始化注意事项”
7	START	A/D 转换起始控制位 0→1→0: 开始; 0→1: A/D 转换复位并且置 $\overline{\text{EOC}}$ 为“1”

ADCR (26H) 寄存器

A/D 转换控制寄存器用来控制 A/D 转换。ADCR 的第 2~0 位用来选择模拟输入通道，总共有 8 个通道可以选择。ADCR 的第 5~3 位用来设置 PB 的工作模式，PB 可以做为模拟输入通道，或是数字输入/输出口，由这 3 位来决定。如果 PB 选择为模拟输入，则其输入/输出功能和上拉电阻将失效，而 A/D 转换电路会被使能。 $\overline{\text{EOC}}$ 位(ADCR 的第 6 位)是 A/D 转换结束标志位。通过检测这个标志位可以知道 A/D 转换是否结束。ADCR 的 START 位用来开启 A/D 转换，给 START 位一个上升沿信号和一个下降沿信号可以开始 A/D 转换。为了确保 A/D 转换顺利完成，START 位应保持为“0”，直到 $\overline{\text{EOC}}$ 位变为“0”(A/D 转换完成信号)。

ACSR 的第 7 位是内部测试用的，用户不能使用。ACSR 的第 1 位和第 0 位用来选择 A/D 转换的时钟来源。

当 START 标志由“0”置为“1”时， \overline{EOC} 也置为“1”。

A/D 转换初始化注意事项：

每次改变模拟通道选择位后都要注意初始化 A/D 转换器，否则 \overline{EOC} 可能处于不确定状态。在模拟通道选择位改变的 10 个指令周期内将 START 置 1 后清 0 来初始化 A/D 转换器。模拟通道选择位都清 0，可以不初始化 A/D。

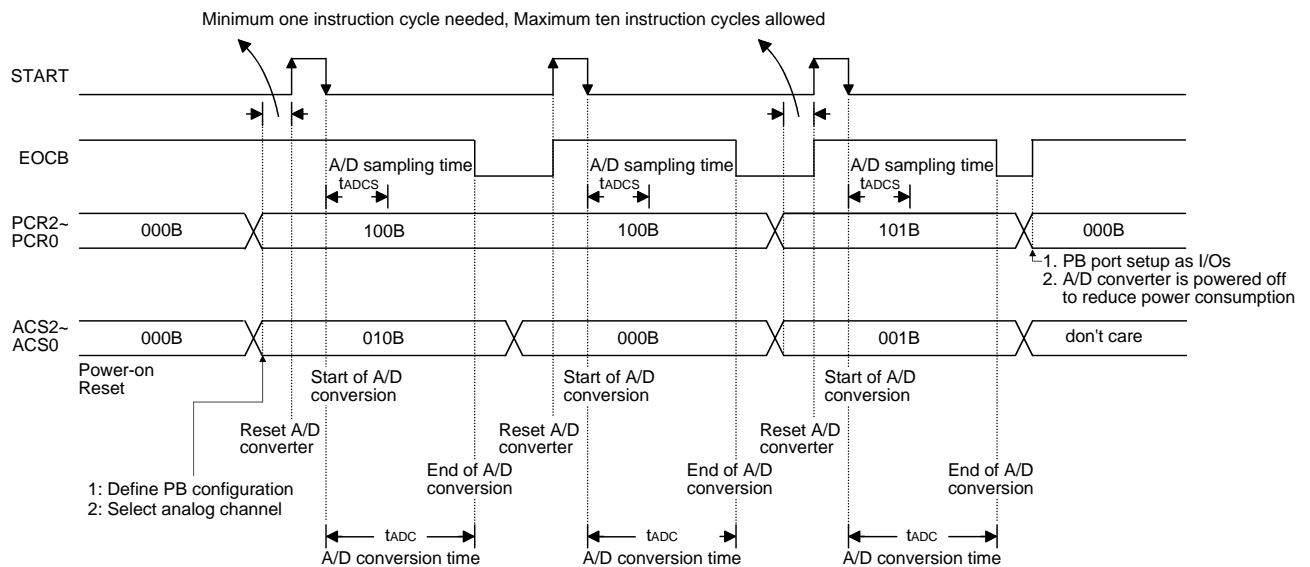
PCR2	PCR1	PCR0	7	6	5	4	3	2	1	0
0	0	0	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
0	0	1	PB7	PB6	PB5	PB4	PB3	PB2	PB1	AN0
0	1	0	PB7	PB6	PB5	PB4	PB3	PB2	AN1	AN0
0	1	1	PB7	PB6	PB5	PB4	PB3	AN2	AN1	AN0
1	0	0	PB7	PB6	PB5	PB4	AN3	AN2	AN1	AN0
1	0	1	PB7	PB6	PB5	AN4	AN3	AN2	AN1	AN0
1	1	0	PB7	PB6	AN5	AN4	AN3	AN2	AN1	AN0
1	1	1	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0

PB 口的设置

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

注：D0~D11 是 A/D 转换结果的低位~高位

ADRL (24H) ADRH (25H) 寄存器



A/D 转换时序图

下面举例说明如何启动和实现 A/D 转换。此例不断扫描 ADCR 寄存器的 $\overline{\text{EOC}}$ 位来判断 A/D 转换是否完成；

例：通过扫描 $\overline{\text{EOC}}$ 位判断 A/D 转换是否完成。

```

mov    a,00000001B
mov    ACSR,a          ; 设置ACSR寄存器，选择fSYS/8做为A/D转换时钟
mov    a,00100000B    ; 在ADCR寄存器中设置PB0~PB3做为A/D输入
mov    ADCR,a         ; 设置AN0进行A/D转换
      :
      :
      :                ; 当模拟通道选择位改变后，START信号（0-1-0）必须在10个
      :                ; 指令周期内发出

```

Start_conversion:

```

clr    START
set    START          ; 复位A/D转换器
clr    START          ; 开始A/D转换

```

Polling_EOC:

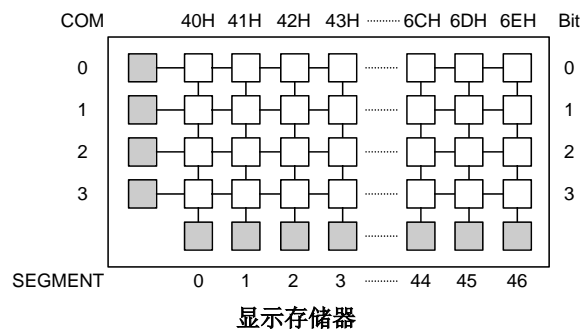
```

sz     EOC            ; 扫描ADCR寄存器的 $\overline{\text{EOC}}$ 位判断A/D转换是否完成
jmp    polling_EOC   ; 继续扫描
mov    a,ADRH        ; 从ADRH寄存器读取A/D转换结果的高位字节
mov    adrh_buffer,a ; 将结果放入用户定义的寄存器中
mov    a,ADRL        ; 从ADRL寄存器读取A/D转换结果的低位字节
mov    adrl_buffer,a ; 将结果放入用户定义的寄存器中
      :
      :
jmp    start_conversion ; 开始下一次 A/D 转换

```

LCD 显示存储器

HT46xU66 为 LCD 显示提供一个嵌入式数据存储器区域。这个区域位于第一段数据存储器(RAM Bank 1)的 40H 到 6EH 单元。存储器段指针 Bank Pointer(BP; RAM 的 04H 单元)是通用存储器 LCD 显示存储器之间切换的开关。当 BP 被置“1”，任何数据写入 40H~6EH(用 MP1 和 R1 间接寻址访问)将会影响 LCD 的显示。当 BP 被清“0”，任何数据写入 40H~6EH 意味着访问一般意义上的数据存储器。LCD 显示存储器能被读出和写入，但是只能通过间接寻址模式，并使用 MP1 来进行。当数据被写入显示数据区域，这些数据自动地被 LCD 驱动器读取来产生相应的 LCD 驱动信号。把“1”或“0”写入显示存储器的相应位，可以控制显示或不显示。下图为显示存储器和 LCD 显示模块之间的映射关系。



LCD 驱动输出

HT46xU66 LCD 驱动器的输出数目可以由配置选项确定为 47×2、47×3 或 46×4(即 1/2、1/3 或 1/4 占空比)。LCD 驱动器偏压产生方式可以分为“R”型或“C”型。如果选为“R”型，不需要外接电容器；如果为“C”型，需要在 C1 和 C2 之间外接一个电容器。LCD 驱动器偏置电压值可以由配置选项设置为 1/2 bias 或 1/3 bias。如果选择为 1/2 bias，则引脚 V2 到地需要接一个电容；如果选择为 1/3 bias 的话，则需要两个电容到地分别地接到引脚 V1、V2。

LCD Segment 输出和逻辑输出

SEG0~SEG23 可配置选择为逻辑输出。一旦 LCD 设置为逻辑输出，LCD 存储区的 bit0 将控制相关 Segment 口的输出状况。

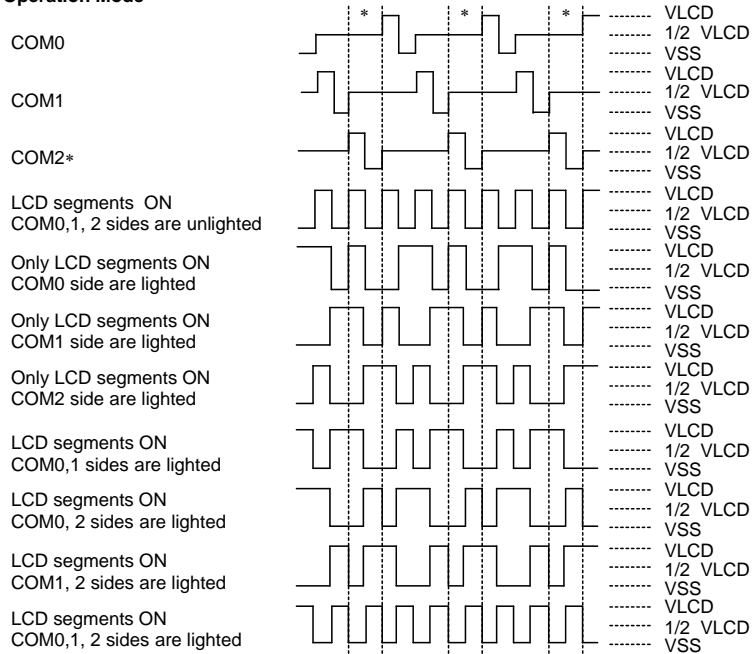
配置选择时 SEG0~SEG7 和 SEG8~SEG15 是按字节设置的，SEG16~SEG23 是按位设置的。

LCD 偏压方式	R 型偏压		C 型偏压	
LCD 偏压类型	1/2bias	1/3bias	1/2bias	1/3bias
V_{MAX}	如果 $V_{DD} > V_{LCD}$ ，则 V_{MAX} 接到 V_{DD} ，反之 V_{MAX} 接到 V_{LCD}		如果 $V_{DD} > \frac{3}{2} V_{LCD}$ ，则 V_{MAX} 接到 V_{DD} ，反之 V_{MAX} 接到 V1	

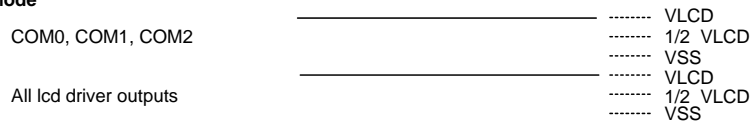
During a Reset Pulse



Normal Operation Mode

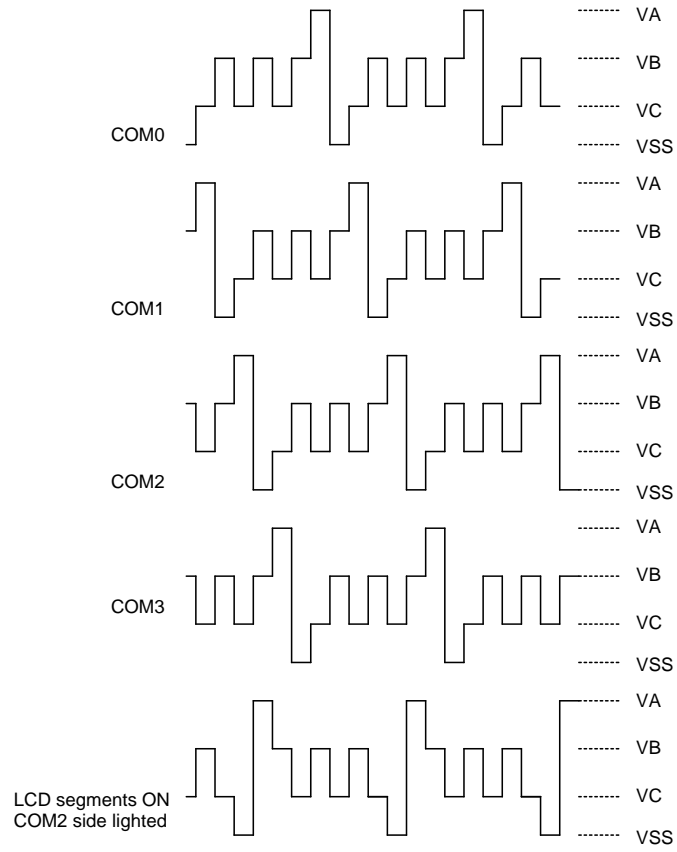


HALT Mode



Note: "*" Omit the COM2 signal, if the 1/2 duty LCD is used.

LCD 驱动输出(1/3duty, 1/2bias, R/C 偏压)



Note: 1/4 duty, 1/3 bias, C type: "VA" 3/2 VLCD, "VB" VLCD, "VC" 1/2 VLCD
 1/4 duty, 1/3 bias, R type: "VA" VLCD, "VB" 2/3 VLCD, "VC" 1/3 VLCD

LCD 驱动输出

低电压复位/检测功能

系统具有低电压检测(LVD)和低电压复位(LVR)功能，可由配置选项设置为打开或关闭。如果选择 LVD 功能，用户可以通过 RTCC.3 来打开/关闭低电压检测，通过 RTCC.5 来读取低电压检测的状态。否则，低电压检测无效。

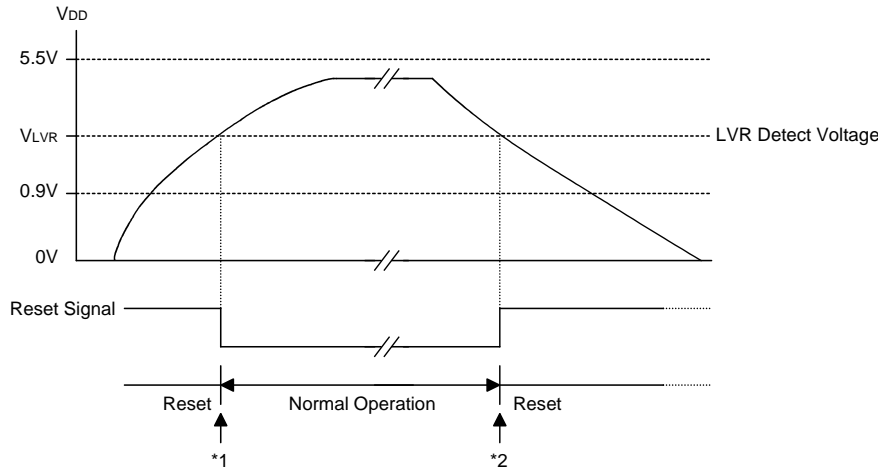
LVR 与外部复位信号有相同的功能，都会使芯片复位。在 HALT 状态下，但是 LVR 是没有作用的。

为了监控器件的工作电压，HT46xU66 提供低电压复位功能。如果器件的工作电压在 $0.9V \sim V_{LVR}$ 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 功能说明如下：

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

V_{DD} 与 V_{LVR} 之间的关系如下所示：



低电压复位

注：*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。

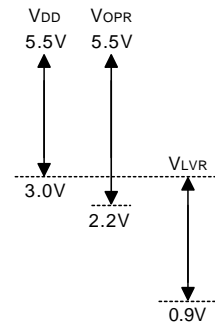
*2：因为低电压状态必须保持 1ms 以上，因此进入复位模式就要有 1ms 的延迟。

RTCC 寄存器的定义如下表：

位	标号	功能
0~2	RT0~RT2	通过控制 8 选 1 多路器输入来选择实时钟预置寄存器的分频输出比例
3	LVDC*	低电压检测打开/关闭(1/0)
4	QOSC	32768Hz 晶振快速起振 0/1：快速/慢速
5	LVDO	低电压检测输出(1/0)，1：检测到低电压，只读
6,7	—	未定义，读数为 0

RTCC (09H) 寄存器

注：“*”一旦 LVD 功能使能，参考发生器应使能，否则参考产生器由 LVR ROM 代码选项控制。LVR 和 LVD 选项与 LVDC 的关系如上表所示。



注： V_{OPR} 是在系统时钟为 4MHz 时，使得芯片正常运行的电压值

异步串行口 — UART

HT46xU66 具有一个全双工的异步串行通信口，可以很方便的与其它具有串行口的芯片通讯。UART 具有许多功能特性，发送或接收串行数据时，将数据组成一个 8 位或 9 位的数据块，连同数据特征位一并传输。当数据过多或数据特征位不正确时，UART 可以检测出错误。UART 功能占用一个内部中断向量，当接收到数据或数据发送结束，触发 UART 中断。

• UART 特性

UART 具有以下功能：

- 全双工异步传输
- 可选择 8 位或 9 位传输格式
- 可选择奇校验、偶校验或无校验
- 可选择 1 位或 2 位停止位
- 8 位预分频的波特率发生器
- 奇偶、帧、噪声和过速检测
- 支持中断和地址检测（最后一位=1）
- 独立的发送和接收允许
- 两层 FIFO 接收缓冲器
- 发送和接收中断
- 下列条件可触发中断
 - 发送完成
 - 发送寄存器空闲
 - 接收完成
 - 过速错误
 - 地址匹配

• UART 外部引脚

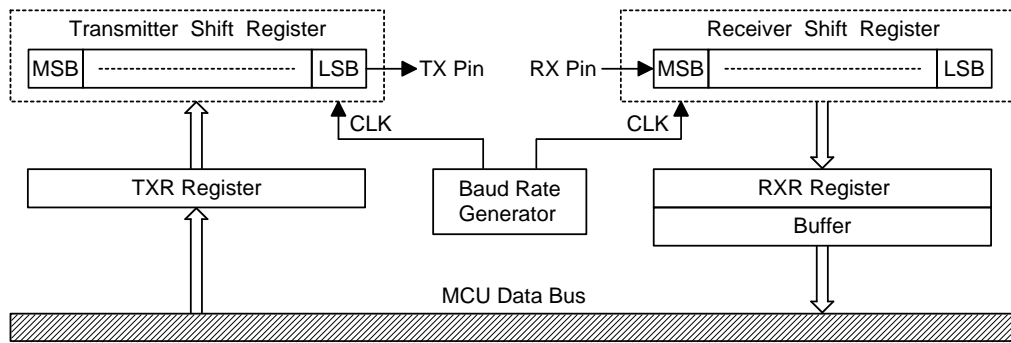
UART 是通过两个外部引脚 TX 和 RX 与外部芯片通讯。TX 引脚是 UART 的发送引脚。当 UCR2 寄存器的 TXEN 位清零，UART 发送功能被禁止，则 TX 引脚可作为普通 IO 口使用。RX 引脚是 UART 的接收引脚。同样的，当 UCR2 寄存器的 RXEN 位清零，UART 接收功能被禁止，则 RX 引脚可作为普通 IO 口使用。若 UARTEN、TXEN 和 RXEN 置位，这些 IO 口将自动转换成相应 TX 输出和 RX 输入，并且 RX 脚的上拉电阻将无效。

• 数据发送

下图显示了 UART 的整体结构。需要发送的数据首先写入 TXR 寄存器，然后在波特率发生器的控制下将寄存器中数据一位位地移到 TX 引脚上，低位在前。TXR 寄存器被映像到单片机的数据存储器中，而发送移位寄存器没有实际地址，所以发送移位寄存器不可直接操作。

数据在波特率发生器的控制下，低位在前高位在后，从外部引脚 RX 进入接收移位寄存器。当数据接收完成，数据从接收移位寄存器移入可被用户程序操作的 RXR 寄存器中。RXR 寄存器被映像到单片机数据存储器中，而接收移位寄存器没有实际地址，所以接收移位寄存器不可直接操作。

需要注意的是，上述发送寄存器 TXR 和接收寄存器 RXR，其实是共享一个地址的数据寄存器 TXR/RXR 寄存器。



UART 数据发送接收图

• **UART 状态控制寄存器**

有 5 个寄存器与 UART 功能相关。寄存器 USR、UCR1 和 UCR2 全面控制 UART，而寄存器 BRG 控制波特率，发送和接收数据则通过寄存器 TXR/RXR。

• **USR 寄存器**

寄存器 USR 是 UART 的状态寄存器，可以通过程序读取。所有 USR 位是只读的。

详细解释如下：

• **TXIF**

TXIF 是发送数据寄存器为空标志。若 TXIF=0，数据还没有从缓冲器加载到移位寄存器中；若 TXIF=1，数据已从缓冲器中加载到移位寄存器中。读取 USR 寄存器再写 TXR 寄存器将清除 TXIF。当 TXEN 被置位，由于发送缓冲器未滿，TXIF 也会被置位。

• **TIDLE**

TIDLE 是数据发送完成标志位。若 TIDLE=0，数据传输中。当 TXIF=1 且数据发送完毕或者暂停字被发送时，TIDLE 置位。TIDLE=1，TX 引脚空闲。读取 USR 寄存器再写 TXR 寄存器将清除 TIDLE 位。

• **RXIF**

RXIF 是接收寄存器状态标志。当 RXIF=0，RXR 寄存器为空；当 RXIF=1，RXR 寄存器接收到新数据。当数据从移位寄存器中加载到 RXR 寄存器中，如果 UCR2 寄存器中的 RIE=1，则会触发中断。当接收数据时检测到一个或多个错误时，相应的标志位 NF、FERR 或 PERR 会在同一周期内置位。读取 USR 寄存器再读 RXR 寄存器，如果 RXR 寄存器中没有新的数据，那么将清除 RXIF 标志。

• **RIDLE**

RIDLE 是接收状态标志。若 RIDLE=0，正在接收数据；若 RIDLE=1，接收器空闲。在接收到停止位和下一个数据的起始位之间，RIDLE 被置位，表明 UART 空闲。

• **OERR**

OERR 是超速错误标志，表示接收缓冲器是否溢出。若 OERR=0，没有数据溢出；若 OERR=1，发生了超速错误，它将影响下一组数据的接收。先读取 USR 寄存器再读 RXR 寄存器将清除此标志位。

• **FERR**

FREE 是帧错误标志位。若 FREE=0，没有帧错误发生；若 FREE=1，当前的数据发生了帧错误。任何复位都会清除该标志位，也可以先读取 USR 寄存器再读 RXR 寄存器来清除此位。

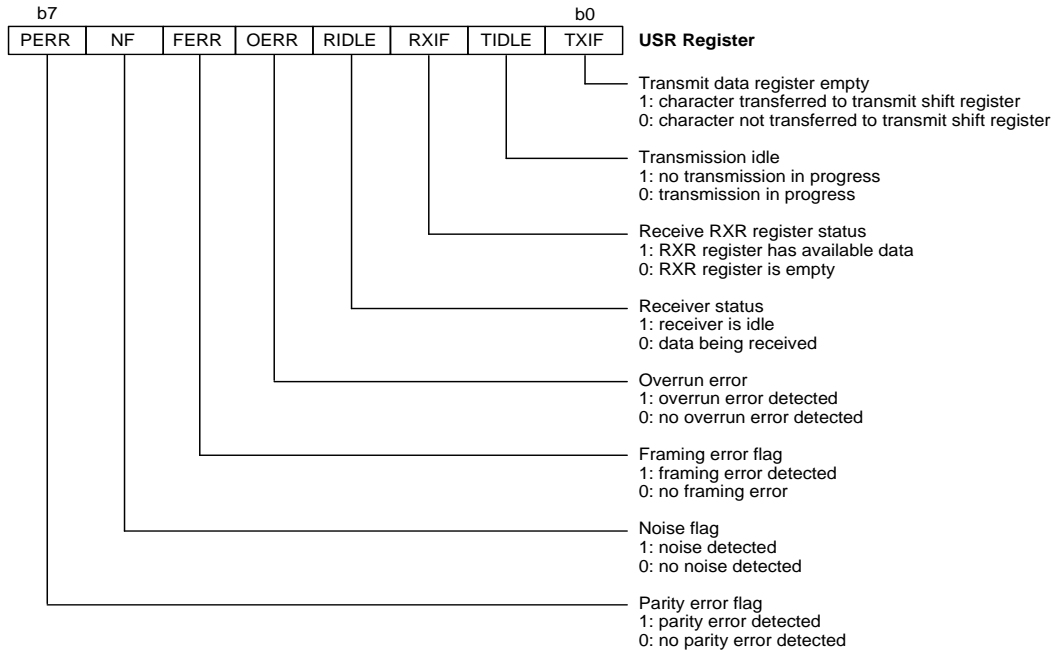
• **NF**

NF 是噪声干扰标志。若 NF=0，没有受到噪声干扰；若 NF=1，UART 接收数据时受到噪声干扰。它与 RXIF 在同周期内置位，但不会与超速标志位同时置位。先读取 USR 寄存器再读 RXR

寄存器将清除此标志位。

· PERR

PERR 是奇偶校验出错标志。若 PERR=0，奇偶校验正确；若 PERR=1，接收到的数据奇偶校验出错。只有使能了奇偶校验此位才有效。任何复位都会清除该标志位，也可以先读取 USR 寄存器再读 RXR 寄存器来清除此位。



· UCR1 寄存器

UCR1 和 UCR2 是 UART 的两个控制寄存器，用来定义各种 UART 功能，例如 UART 的使能与除能、奇偶校验控制和传输数据的长度等等。

详细解释如下：

· TX8

此位只有在传输数据为 9 位的格式中有效，用来存储发送数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

· RX8

此位只有在传输数据为 9 位的格式中有效，用来存储接收数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

· TXBRK

TXBRK 是暂停字发送控制位。TXBRK=0，没有暂停字要发送，TX 引脚正常操作；TXBRK=1，将会发送暂停字，发送器将发送逻辑 0。若 TXBRK 为高，缓冲器中数据发送完毕后，发送器将至少保持 13 位宽的低电平直至 TXBRK 复位。

· STOP

此位用来设置停止位的长度。STOP=1，有两位停止位；STOP=0，只有一位停止位。

· PRT

奇偶校验选择位。PRT=1，奇校验；PRT=0，偶校验。

· PREN

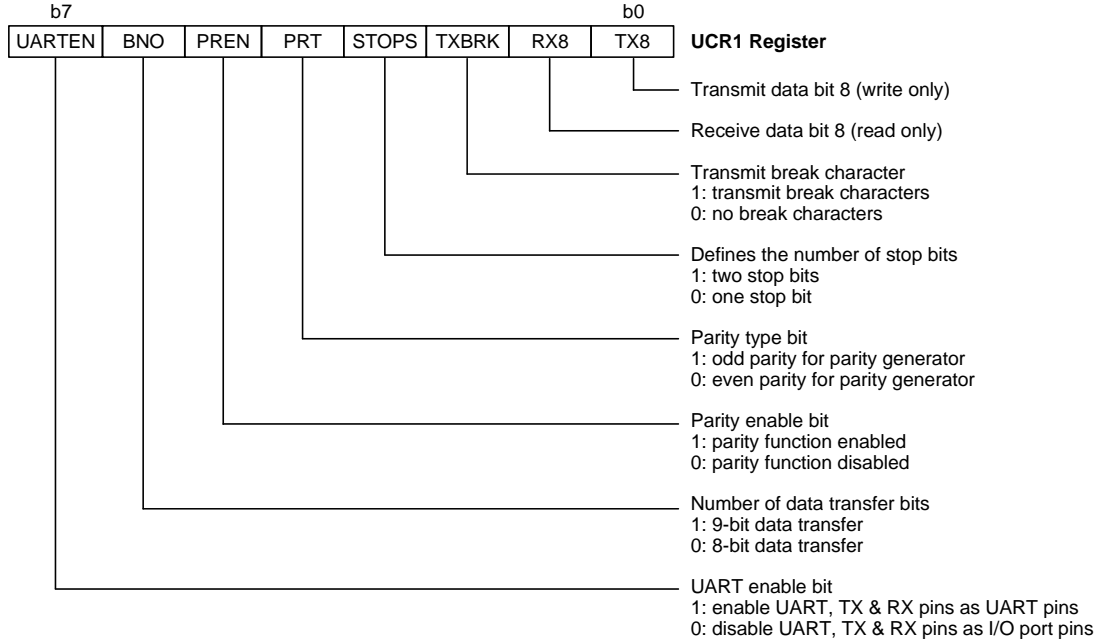
此位为奇偶校验使能位。PREN=1，使能奇偶校验；PREN=0，除能奇偶校验。

· BNO

BNO 是发送位数控制位。BNO=1，传输数据为 9 位；BNO=0，传输数据为 8 位。若选择了 9 位数据传输格式，RX8 和 TX8 将分别存储接收和发送数据的第 9 位。

· UARTEN

此位为 UART 的使能位。UARTEN=0, UART 除能, RX 和 TX 可用作普通的输入输出; UARTEN=1, UART 使能, TX 和 RX 将分别由 TXEN 和 RXEN 控制。当 UART 被除能将清除缓冲器, 所有缓冲器中的数据将被忽略, 另外波特率计数器、错误和状态标志位被复位, TXEN、RXEN、TXBRK、RXIF、OERR、FERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位, UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零, 所有发送和接收将停止, 模块也将复位成上述状态。当 UART 再次使能时, 它将在上次配置下重新工作。



• **UCR2 寄存器**

UCR2 是 UART 的另一个控制寄存器, 它的主要功能是使能或除能发送和接收允许以及 UART 的各种中断源。它也可用来控制波特率, 使能接收唤醒和地址侦测。

详细解释如下:

• **TEIE**

此位为发送寄存器为空中断的使能或除能位。若 TEIE=1, 当 TXIF 置位时, UART 的中断请求标志置位; 若 TEIE=0, UART 中断请求标志不受 TXIF 的影响。

• **TIE**

此位为发送器空闲中断的使能或除能位。若 TIE=1, 当 TIDLE 置位时, UART 的中断请求标志置位; 若 TIE=0, UART 中断请求标志不受 TIDLE 的影响。

• **RIE**

此位为接收中断使能或除能位。若 RIE=1, 当 OERR 或 RXIF 置位时, UART 的中断请求标志置位; 若 RIE=0, UART 中断请求标志不受 OERR 和 RXIF 影响。

• **WAKE**

此位为接收唤醒功能的使能和除能位。若 WAKE=1 且在暂停模式下, RX 引脚的下降沿将唤醒单片机。若 WAKE=0 且在暂停模式下, RX 引脚的任何边沿都不能唤醒单片机。

• **ADDEN**

此位为地址检测使能和除能位。ADDEN=1, 地址检测使能, 此时数据的第 8 位 (BON=0) 或第 9 位 (BON=1) 为高, 那么接到的是地址而非数据。若相应的中断使能且接收到的值最高位为 1, 那么中断请求标志将会被置位, 若最高位为 0, 那么将不会产生中断且收到的数据也会被忽略。

• **BRGH**

此位为波特率发生器高低速选择位, 它和 BRG 寄存器一起控制 UART 的波特率。BRGH=1,

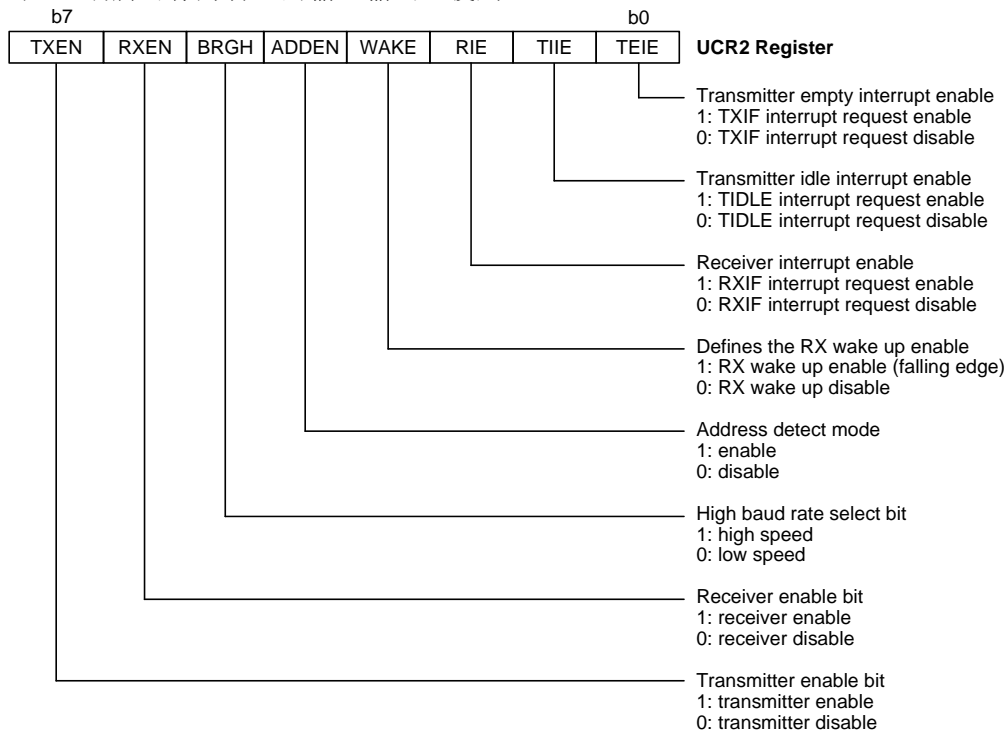
为高速模式；BRGH=0，为低速模式。

· RXEN

此位为接收使能位。RXEN=0，接收将被除能，接收器立刻停止工作。另外缓冲器将被复位，此时RX引脚可作普通的输入输出端口使用。若RXEN=1且UARTEN=1，则接收将被使能，RX引脚将由UART来控制。在数据传输时清除RXEN将中止数据接收且复位接收器，此时RX引脚可作为普通输入输出端口使用。

· TXEN

此位为发送使能位。TXEN=0，发送将被除能，发送器立刻停止工作。另外缓冲器将被复位，此时TX引脚可作为普通的输入输出端口使用。若TXEN=1且UARTEN=1，则发送将被使能，TX引脚将由UART来控制。在数据传输时清除TXEN将中止数据发送且复位发送器，此时TX引脚可作为普通的输入输出端口使用。



• 波特率发生器

UART自身具有一个波特率发生器，通过它可以设定数据传输速率。波特率是由一个独立的内部8位计数器产生，它由BRG寄存器和UCR2寄存器的第2位BRGH来控制。BRGH是决定波特率发生器处于高速模式还是低速模式，从而决定计算公式的选用。BRG寄存器的值N可根据下表中的公式计算，N的范围是0到255。

UCR2的BRGH位	0	1
波特率	$\frac{f_{SYS}}{64(N+1)}$	$\frac{f_{SYS}}{16(N+1)}$

为得到相应的波特率，首先需要设置BRGH来选择相应的计算公式从而算出BRG的值。由于BRG的值不连续，所以实际波特率和理论值之间有一个偏差。下面举例怎样计算BRG寄存器中的值N和误差。

· 波特率和误差的计算

系统选用 8M 晶振且 BRGH=0，若期望的波特率为 9600，计算它的 BRG 寄存器的值 N，实际波特率和误差。

$$\text{根据上表, 波特率 } BR = \frac{f_{SYS}}{64(N+1)}$$

$$\text{转换后的公式 } N = \frac{f_{SYS}}{BR * 64} - 1$$

$$\text{代入参数 } N = \frac{8000000}{9600 * 64} - 1 = 12.0208$$

取最接近的值，十进制 12 写入 BRG 寄存器，实际波特率如下

$$BR = \frac{8000000}{64(12+1)} = 9615$$

$$\text{误差 } \frac{9615 - 9600}{9600} = 0.16\%$$

下面两表给出 BRGH 取不同值时的实际波特率和误差。

波特率 K/BPS	BRGH=0											
	f _{SYS} =8MHz			f _{SYS} =7.159MHz			f _{SYS} =4MHz			f _{SYS} =3.579545MHz		
	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error
0.3	-	-	-	-	-	-	207	0.300	0.00	185	0.300	0.00
1.2	103	1.202	0.16	92	1.203	0.23	51	1.202	0.16	46	1.19	-0.83
2.4	51	2.404	0.16	46	2.38	-0.83	25	2.404	0.16	22	2.432	1.32
4.8	25	4.807	0.16	22	4.863	1.32	12	4.808	0.16	11	4.661	-2.9
9.6	12	9.615	0.16	11	9.322	-2.9	6	8.929	-6.99	5	9.321	-2.9
19.2	6	17.857	-6.99	5	18.64	-2.9	2	20.83	8.51	2	18.643	-2.9
38.4	2	41.667	8.51	2	37.29	-2.9	1	-	-	1	-	-
57.6	1	62.5	8.51	1	55.93	-2.9	0	62.5	8.51	0	55.93	-2.9
115.2	0	125	8.51	0	111.86	-2.9	-	-	-	-	-	-

BRGH=0 时的波特率和误差

波特率 K/BPS	BRGH=1											
	f _{SYS} =8MHz			f _{SYS} =7.159MHz			f _{SYS} =4MHz			f _{SYS} =3.579545MHz		
	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error	BRG	Kbaud	Error
0.3	-	-	-	-	-	-	-	-	-	-	-	-
1.2	-	-	-	-	-	-	207	1.202	0.16	185	1.203	0.23
2.4	207	2.404	0.16	185	2.405	0.23	103	2.404	0.16	92	2.406	0.23
4.8	103	4.808	0.16	92	4.811	0.23	51	4.808	0.16	46	4.76	-0.83
9.6	51	9.615	0.16	46	9.520	-0.832	25	9.615	0.16	22	9.727	1.32
19.2	25	19.231	0.16	22	19.454	1.32	12	19.231	0.16	11	18.643	-2.9
38.4	12	38.462	0.16	11	37.287	-2.9	6	35.714	-6.99	5	37.286	-2.9
57.6	8	55.556	-3.55	7	55.93	-2.9	3	62.5	8.51	3	55.930	-2.9
115.2	3	125	8.51	3	111.86	-2.9	1	125	8.51	1	111.86	-2.9
250	1	250	0	-	-	-	0	250	0	-	-	-

BRGH=1 时的波特率和误差

UART 设置与控制

简介

UART 采用标准的不归零码传输数据，这种方法通常被称为 NRZ 法。它由 1 位起初位，8 位或 9 位数据位和 1 位或者两位停止位组成。奇偶校验是由硬件自动完成的，可设置成奇校验、偶校验和无校验三种格式。常用的数据传输格式由 8 位数据位，1 位停止位，无校验组成，用 8, N, 1 表示，它是系统上电的默认格式。数据位数、停止位数和奇偶校验由 UCR1 寄存器的 BNO、PRT、PREN 和 STOPS 设定。用于数据发送和接收的波特率由一个内部的 8 位定时器产生，数据传输时低位在前高位在后。尽管 UART 发送器和接收器在功能上相互独立，但它们使用相同的数据传输格式和波特率，在任何情况下，停止位是必须的。

UART 的使能和除能

UART 是由 UCR1 寄存器的 UARTEN 位来使能和除能的。它的发送引脚 TX 和接收引脚 RX 分别与 PC6 和 PC7 复用，UARTEN 的一个基本功能就是控制这两个引脚。若 UARTEN、TXEN 和 RXEN 都为高，则 PC6 和 PC7 分别为 UART 的发送端口和接收端口，而不能作为普通的输入输出端口使用。若没有数据发送，TX 引脚默认状态为高电平。

UARTEN 清零将除能 TX 和 RX，使其可作为普通的输入输出端口使用。当 UART 被除能清除缓冲器，所有缓冲器中的数据将被忽略，另外错误和状态标志位被复位，TXEN、RXEN、TXBRK、RXIF、OERR、FERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

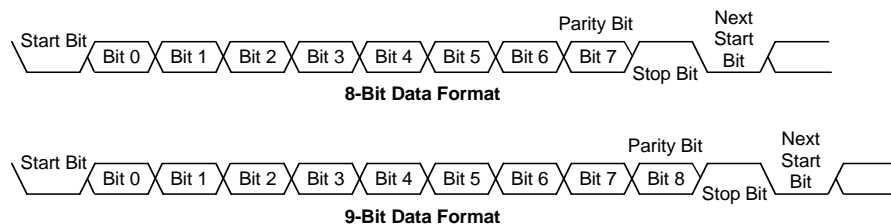
数据位、停止位位数以及奇偶校验的选择

数据传输格式由数据长度、是否校验、校验类型地址表明位以及停止位长度组成。它们都是由 UCR1 寄存器的各个位控制的。BNO 决定数据传输是 8 位还是 9 位；PRT 决定校验类型；PRTEN 决定时是否选择奇偶校验；而 STOPS 决定选用 1 位还是 2 位停止位。下表列出了各种数据传输格式。地址表明位用来确定此帧是否为地址。

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bits
8 位数据位				
1	8	0	0	1
1	7	0	1	1
1	7	1 ¹	0	1
9 位数据位				
1	9	0	0	1
1	8	0	1	1
1	8	1 ¹	0	1

发送和接收数据格式

下图是传输 8 位和 9 位数据的波形。



· UART 发送器

UCR1 寄存器的 BNO 位是控制数据传输的长度。BNO=1 其长度为 9 位，第 9 位 MSB 存储在 UCR1 寄存器的 TX8 中。发送器的核心是发送移位寄存器 TSR，它的数据由发送寄存器 TXR 提供，应用程序只须将发送数据写入 TXR 寄存器。上组数据的停止位发出前，TSR 寄存器禁止写入。如果还有新的数据要发送，一旦停止位发出，待发数据将会从 TXR 寄存器加载到 TSR 寄存器。TSR 不像其它寄存器一样映射在数据存储区，所以应用程序不能对其进行读写操作。TXEN=1，发送使能，但若 TXR 寄存器没有数据或者波特率没有设置，发送器将不会工作。先写 TXR 寄存器再 TXEN 也会触发发送。当发送器使能，若 TSR 寄存器为空，数据写入 TXR 寄存器将会直接加载到 TSR 寄存器中。发送器工作时，TXEN 清零，发送器将立刻停止工作并且复位，此时 TX 引脚可作为普通的输入输出使用。

· 发送数据

当 UART 发送数据时，数据从移位寄存器中移到 TX 引脚上，其低位在前高位在后。在发送模式中，TXR 寄存器在内部总线和发送移位寄存器间形成一个缓冲。如果选择 9 位数据传输格式，最高位 MSB 存储在 UCR1 寄存器的 TX8 中。

发送器初始化可由如下步骤完成：

--正确地设置 BNO、PRT、PREN 和 STOPS 位以确定数据长度、校验类型和停止位长度。

--设置 BRG 寄存器，选择期的波特率。

--置高 TXEN，使引脚作为 UART 的发送端而非普通的输入输出端口。

--读取 USR 寄存器，然后将待发数据写入 TXR 寄存器，此步骤会清除 TXIF 标志位。

--如果要发送多个数据只需重复上一步骤。

当 TXIF=0 时，数据将禁止写入 TXR 寄存器。可以通过以下步骤来清除 TXIF：

1. 读取 USR 寄存器
2. 写 TXR 寄存器

只读标志位 TXIF 由 UART 硬件置位。若 TXIF=1，TXR 寄存器为空，其它数据可以写入而不会覆盖以前的数据。若 TEIE=1，TXIF 标志位会影响中断。

在数据传输时，写 TXR 指令会将待发数据暂存在 TXR 寄存器中，当前数据发送完毕后，待发数据被加载到发送移位寄存器中。当发送器空闲时，写 TXR 指令会将数据直接加载到 TSR 寄存器中，数据传输立刻开始且 TXIF 置位。当一帧数据发送完毕，TIDLE 将被置位。

可以通过以下步骤来清除 TIDLE：

1. 读取 USR 寄存器
2. 写 TXR 寄存器

清除 TXIF 和 TIDLE 软件执行次序相同。

· 发送暂停字

若 TXBRK=1，下一帧将会发送暂停字。它是同一个起始位、13*N (N=1, 2,.....) 位逻辑 0 以及停止位组成。置位 TXBRK 将会发送暂停字，而清除 TXBRK 产生停止位，传输暂停字不会产生中断。需要注意的是，暂停字至少 13 位宽。若 TXBRK 持续为高，那么发送器会一直发送暂停字；当应用程序清除了 TXBRK，发送器将传输最后一帧暂停字再加上一位或者两位停止位。暂停字后的高电平保证下一帧数据起始位的检测。

• UART 接收器

· 简介

UART 接收器支持 8 位或者 9 位数据接收。若 BNO=1，数据长度为 9 位，而最高位 MSB 存放在 UCR1 寄存器的 RX8 中。接收器的核心是串行移位寄存器 RSR。RX 引脚上的数据送入数据恢复器中，它在 16 倍波特率的频率下工作，而串行移位器工作在正常波特率下。当在 RX 引脚上检测到停止位，数据从 RSR 寄存器中加载到 RXR 寄存器。RX 引脚上的每一位数据会被采样三次以判断其逻辑状态。RSR 不像其它寄存器一样映射在数据存储器，所以应用程序不能对其进行读写操作。

· 接收数据

当 UART 接收数据时，数据低位在前高位在后，连续地从 RX 引脚进入。RXR 寄存器在内部总线和接收移位寄存器间形成一个缓冲。RXR 寄存器是一个两层的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 RXR 寄存器，否则忽略第三帧数据并且发生过速错误。

接收器的初始化可由如下步骤完成：

- 正确地设置 BNO、PRT、PREN 和 STOPS 位以确定数据长度、校验类型和停止位长度。
- 设置 BRG 寄存器，选择期的波特率。
- 置高 TXEN，使引脚作为 UART 的发送端而非普通的输入输出端口。

此时接收器被使能并检测起始位。

接收数据将会发生如下事件：

- 当 RXR 寄存器中有一帧以上的数据时，USR 寄存器中的 RXIF 位将会置位。
- 若 RIE=1，数据从 RSR 寄存器加载到 RXR 寄存器中将产生中断。
- 若接收器检没到帧错误、噪声干扰错误、奇偶出错或过速错误，那么相应的错误标志位置位。

可以通过如下步骤来清除 RXIF：

1. 读取 USR 寄存器
2. 读取 RXR 寄存器

· 接收暂停字

UART 接收任何暂停字都会当作帧错误处理。接收器只根据 BNO 和 STOPS 位确定一帧数据的长度。若暂停字数大于 BNO 和 STOPS 位指定的长度，接收器认为接收已完结，RXIF 和 FERR 置位，RXR 寄存器清 0，若相应的中断允许且 RIDLE 为高将会产生中断。若暂停字较长，接收器收到起始位、数据位将会置位 FERR 标志，且在下一起始位前必须检测到有效的停止位。暂停字只会被认为包含信息 0 且会置位 FERR 标志。暂停字将会加载到缓冲器中，在接收到停止位前不会再接收数据，没有检测到停止位也会置位只读标志位 RIDLE。

UART 接收到暂停字会产生以下事件：

- 帧错误标志位 FERR 置位。
- RXR 寄存器清零。
- OERR、NF、PERR、RIDLE 或 RXIF 可能会置位。

· 空闲状态

当 UART 接收数据时，即在起初位和停止位之间，USR 寄存器的接收标志位 RIDLE 清零。在停止位和下一帧数据的起始位之间，RIDLE 被置位，表示接收器空闲。

- 接收中断

USR 寄存器的只读标志位 RXIF 由接收器的边缘触发置位。若 RIE=1，数据从移位寄存器 RSR 加载到 RXR 寄存器时产生中断，同样地，过速也会产生中断。

- 接收错误处理

UART 会产生几种接收错误，下面部分将描述各错误以及怎样处理。

- 过速——OERR 标志

RXR 寄存器是一个两层的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 RXR 寄存器，否则发生过速错误。

产生过速错误时将会发生以下事件：

- USR 寄存器中 OERR 被置位。
- RXR 寄存器中数据不会丢失。
- RSR 寄存器数据将会被覆盖。
- 若 RIE=1，将会产生中断。

- 噪声干扰——NF 标志

数据恢复时多次采样可以有效的鉴别出噪声干扰。当检测到数据受到噪声干扰时将会发生以下事件：

- 在 RXIF 上升沿，USR 寄存器中只读标志位 NF 置位。
- 数据从 RSR 寄存器加载到 RXR 寄存器中。
- 不产生中断，此位置位的同时由 RXIF 请求中断。

先读取 USR 寄存器再读取 RXR 寄存器将复位 NF。

- 帧错误——FERR 标志

若在停止位上检测到 0，USR 寄存器中只读标志 FERR 置位。若选择两位停止位，两停止都必须为高，否则将置位 FERR。它同数据一起存储在缓冲器中，可被任何复位清除。

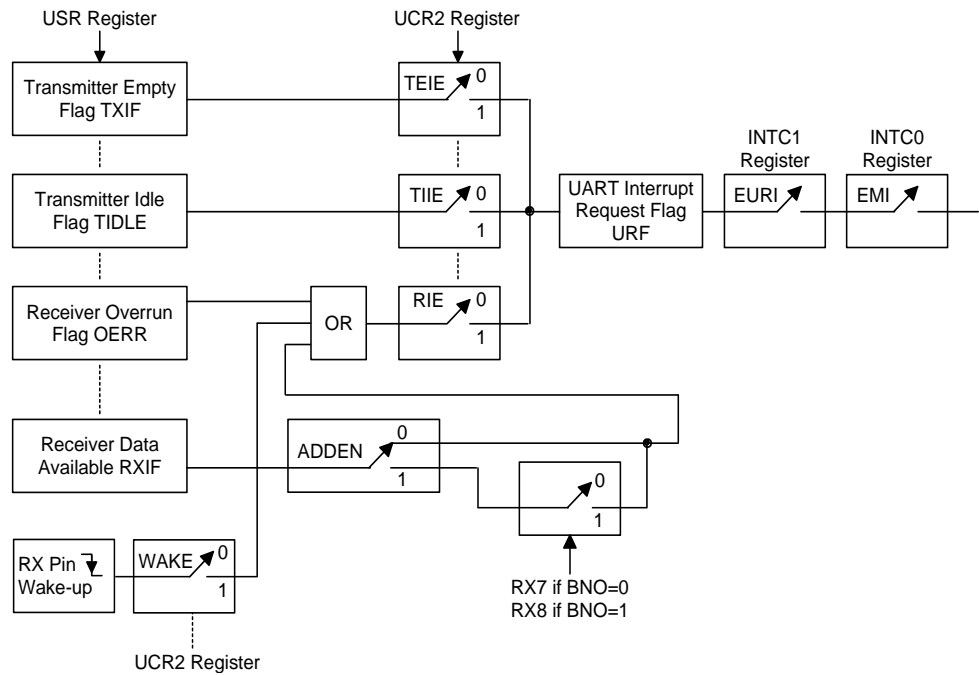
- 奇偶校验错误——PERR 标志

若接收到数据出现奇偶校验错误，USR 寄存器中只读标志 PERR 置位。只有使能了奇偶校验，选择了校验类型，此标志位才有效。它同数据一起存储在缓冲器中，可被任何复位清除。注意，FERR 和 PERR 与相应的数据一起存储在缓冲器中，在读取数据之前必须先访问错误标志位。

- 接收中断图解

UART 拥有单独的内部中断和独立的中断变量。发送寄存器为空、发送器空闲、接收器数据有效、过速和地址检测和 RX 引脚唤醒都会产生中断。若 UART 中断允许且堆栈未满，程序将会跳转到相应的中断向量执行中断服务程序，而后再返回主程序。其中四种，若 UCR2 寄存器中相应中断允许位被置位，USR 寄存器中标志位将会产生中断。发送器有两个相应的中断允许位而接收器共用一个中断允许位。这些允许位可用于禁止个别的 UART 中断源。

地址检测也是 UART 的中断源，它没有相应的标志位，若 UCR2 寄存器中 ADDEN=1，当检测到地址将会产生 UART 中断。RX 引脚唤醒也可以产生 UART 中断，它没有相应的标志位，当 UXR2 中的 WAKE 和 RIE 位被置位，RX 引脚上有下降沿可以唤醒单片机。应注意，RX 唤醒中断发生时，系统必须延时 1024 个系统时钟才能正常工作。



UART 中断框图

• 地址检测模式

置位 UCR2 寄存器中的 ADDEN 将启动地址检测模式。若 ADDEN 有效，只有在接收到数据最高位为 1 才会产生中断，中断允许位 EURI 和 EMI 也要使能才会产生中断。地址的最高位为第 9 位 (BNO=1) 或第 8 位 (BNO=0)，若此位为高，则接收到的是地址而非数据。只有接收的数据的最后位为高才会产生中断。若 ADDEN 除能，每接收到一个有效数据便会置位 RXIF，而不用考虑数据的最后一位。地址检测和奇偶校验在功能上相互排斥，若地址检测模式使能，必须保证操作的正确，同时必须将奇偶检验使能位清零，除能奇偶校验。

ADDEN	位 9 (BNO=1), 位 8 (BNO=0)	产生 UART 中断
0	0	√
	1	√
1	0	×
	1	√

ADDEN 位功能

• 暂停模式下的 UART 功能

当 MCU 进入暂停模式，UART 将停止工作。当芯片进入暂停模式，模块的所有时钟关闭。当 UART 传送数据时，MCU 进入暂停模式，发送将停止并且 TX 引脚保持高电平。同样地，当 MCU 接收数据时进入暂停模式，数据接收也会停止。当单片机进入暂停模式，USR、UCR1 和 UCR2、接收/发送寄存器、BRG 寄存器都不会受到影响。

UART 功能中包括了 RX 引脚的唤醒功能，由 UCR2 寄存器中 WAKE 位控制。进入暂停模式前，若该标志位与 UART 允许位 UARTEN、接收器允许位 RXEN 和接收器中断位 RIE 都被置位，则 RX 引脚的下降沿可唤醒单片机。唤醒后系统需延时 1024 个系统时钟才能正常工作，在此期间，RX 引脚上的任何数据将被忽略。

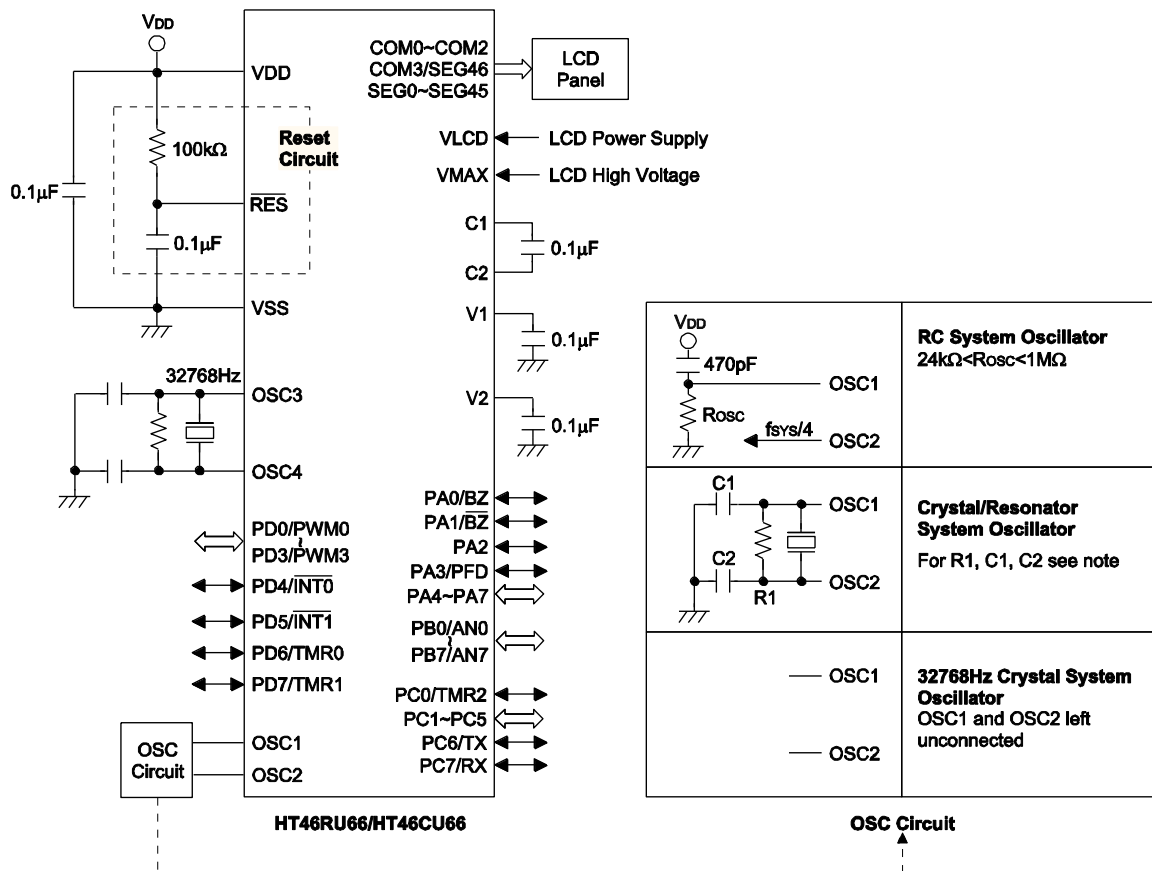
若要唤醒并产生 UART 中断，全局中断允许位 EMI 和 UART 中断允许 EURI 也必须置位；若这两标志位没有被置位，那么，单片机将可以被唤醒但不会产生中断。同样唤醒后系统需延时 1024 个系统时钟才能正常工作，然后才会产生 UART 中断。

配置选项

下表列出了所有配置选项。所有选项必须正确定义，以保证系统正常运行。

配置选项
<p>OSC 类型的选项。 这个选项确定是否选择一个 RC 或晶体或 32768Hz 晶体来作为系统时钟。</p>
<p>WDT, RTC 和时基的时钟源选项。 有三种选择的方式：系统时钟四分频或 RTC OSC 或 WDT OSC。</p>
<p>WDT 打开/关闭选项。 由配置选项，WDT 打开或关闭。</p>
<p>WDT 溢出周期选项。 有四种选择的方式：WDT 时钟源的 $2^{12}/f_s \sim 2^{13}/f_s$、$2^{13}/f_s \sim 2^{14}/f_s$、$2^{14}/f_s \sim 2^{15}/f_s$ 或 $2^{15}/f_s \sim 2^{16}/f_s$ 分频。</p>
<p>CLR WDT 次数选项。这个选项定义用指令清除 WDT 的方法。“One time”指用“CLR WDT”指令功能清除 WDT。“Two times”指的是必须要用 CLR WDT1 和 CLR WDT2 二条指令来清除 WDT。</p>
<p>时基溢出周期选项。 时基溢出周期范围从 $clock/2^{12} \sim clock/2^{15}$。“clock”是由配置选项确定的时钟源频率。</p>
<p>蜂鸣器输出频率选项。有八种输出频率供选择：$clock/2^2 \sim clock/2^9$。“clock”是由配置选项确定的时钟源频率。</p>
<p>Wake-up 选项。 这个选项用来设置唤醒功能。外部的输入/输出引脚(仅 PA 具有)的下降沿，具有将系统从 HALT 模式唤醒的能力。</p>
<p>上拉电阻选项。 这个选项用来设置输入/输出口在输入模式时，是否带有内部上拉电阻。PA、PB 和 PD 可以独立设置(按位设置)。</p>
<p>输入/输出与其它功能共用引脚选项。 PA0/BZ、PA1/BZ：PA0 和 PA1 可以设置为一般输入/输出口或蜂鸣器输出。 PA3/PFD：PA 可以设置为一般输入/输出口或 PFD 输出。</p>
<p>LCD Common 端选项。 有三种选择：2Common(1/2 duty)，或 3Common(1/3 duty)，或 4Common(1/4 duty)。如果选择了 4 Common，那么段输出引脚“SEG46”将被作为 Common 端输出。</p>
<p>LCD 偏压选项。有二种选择：1/2 bias 或 1/3 bias。</p>
<p>LCD 偏压方式选项。这个选项确定是 R 型偏压还是 C 型偏压。</p>
<p>LCD 驱动器时钟源选项。 有七种频率信号可选择：$fs/2^2 \sim fs/2^8$。“fs”是由配置选项确定的时钟源频率。</p>
<p>LCD 在 HALT 模式开关选项。</p>
<p>LCD Segment 做为逻辑输出选项。(字节、字节、位、位、位、位、位、位、位选择) [SEG0~SEG7]、[SEG8~SEG15]、SEG16、SEG17、SEG18、SEG19、SEG20、SEG21、SEG22、SEG23</p>
<p>LVR 选项。LVR 打开或关闭。</p>
<p>LVD 选项。LVD 打开或关闭。</p>
<p>PFD 选项。 如果 PA3 被选作为 PFD 输出，有二种选择；一种是 PFD0 作为 PFD 输出，另一种是 PFD1 作为 PFD 输出。PFD0、PFD1 分别是定时/计数器 0 和定时/计数器 1 的定时器溢出信号。</p>
<p>PWM 选项：(7+1)或(6+2)模式 PD0：电平输出或 PWM0 输出 PD1：电平输出或 PWM1 输出 PD2：电平输出或 PWM2 输出 PD3：电平输出或 PWM3 输出</p>
<p>INT0 或 INT1 触发方式选项：禁止，上升沿触发，下降沿触发，或两者皆可触发</p>

应用电路



下表是不同晶体频率时，C1、C2 和 R1 的不同取值。

晶体或共振器	C1、C2	R1
4MHz 晶体	0pF	10kΩ
4MHz 共振器	10pF	12kΩ
3.58MHz 晶体	0pF	10kΩ
3.58MHz 共振器	25pF	10kΩ
2MHz 晶体和共振器	25pF	10kΩ
1MHz 晶体	35pF	27kΩ
480kHz 共振器	300pF	9.1kΩ
455kHz 共振器	300pF	10kΩ
429kHz 共振器	300pF	10kΩ

R1 的作用是在低电压的时候确保关闭振荡，此低电压值低于单片机的最低工作电压。需要注意的是如果 LVR 使能，可以不加 R1。

注：电阻和电容值选取的原则是使 VDD 保持稳定并在 \overline{RES} 置为高以前把工作电压保持在允许的范围內。
“*”为了避免噪声干扰，连接 RES 引脚的线请尽可能地短

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	2	无
SZ	[m]	1 ⁽²⁾	无
SZA	[m]	1 ⁽²⁾	无
SZ	[m].i	1 ⁽²⁾	无
SNZ	[m].i	1 ⁽²⁾	无
SIZ	[m]	1 ⁽³⁾	无
SDZ	[m]	1 ⁽³⁾	无
SIZA	[m]	1 ⁽²⁾	无
SDZA	[m]	1 ⁽²⁾	无
CALL	addr	2	无
RET		2	无
RET	A,x	2	无
RETI		2	无
查表			
TABRDC	[m]	2 ⁽¹⁾	无
TABRDL	[m]	2 ⁽¹⁾	无
其它指令			
NOP		1	无
CLR	[m]	1 ⁽¹⁾	无
SET	[m]	1 ⁽¹⁾	无
CLR	WDT	1	TO,PDF
CLR	WDT1	1	TO ⁽⁴⁾ ,PDF ⁽⁴⁾
CLR	WDT2	1	TO ⁽⁴⁾ ,PDF ⁽⁴⁾
SWAP	[m]	1 ⁽¹⁾	无
SWAPA	[m]	1	无
HALT		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： $ACC \leftarrow ACC + [m] + C$
 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： $[m] \leftarrow ACC + [m] + C$
 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： $ACC \leftarrow ACC + [m]$
 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： $ACC \leftarrow ACC + x$
 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： $[m] \leftarrow ACC + [m]$
 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： $ACC \leftarrow ACC \text{ “AND” } [m]$
 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： $ACC \leftarrow ACC \text{ “AND” } x$
 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： $[m] \leftarrow ACC \text{ “AND” } [m]$
 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： $Stack \leftarrow Program\ Counter + 1$
 $Program\ Counter \leftarrow addr$
 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： $[m] \leftarrow 00H$
 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： $[m].i \leftarrow 0$
 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： $WDT \leftarrow 00H$
 $PDF \& TO \leftarrow 0$
 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： $WDT \leftarrow 00H^*$

$PDF \& TO \leftarrow 0^*$

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： $WDT \leftarrow 00H^*$

$PDF \& TO \leftarrow 0^*$

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： $[m] \leftarrow \overline{[m]}$

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： $ACC \leftarrow \overline{[m]}$

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 $ACC.3 \sim ACC.0 > 9$ 或 $AC=1$
 那么 $[m].3 \sim [m].0 \leftarrow (ACC.3 \sim ACC.0) + 6$, $AC1 = \overline{AC}$
 否则 $[m].3 \sim [m].0 \leftarrow (ACC.3 \sim ACC.0)$, $AC1 = 0$
 并且
 如果 $ACC.7 \sim ACC.4 + AC1 > 9$ 或 $C=1$
 那么 $[m].7 \sim [m].4 \leftarrow (ACC.7 \sim ACC.4) + 6 + AC1$, $C=1$
 否则 $[m].7 \sim [m].4 \leftarrow (ACC.7 \sim ACC.4) + AC1$, $C=C$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

DEC [m] 数据存储器内容减 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： $[m] \leftarrow [m] - 1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DECA [m] 数据存储器内容减 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： $ACC \leftarrow [m] - 1$
 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：
 $Program\ Counter \leftarrow Program\ Counter + 1$
 $PDF \leftarrow 1$
 $TO \leftarrow 0$

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： **Program Counter ← 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: $\text{Program Counter} \leftarrow \text{Program Counter} + 1$
 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: $\text{ACC} \leftarrow \text{ACC} \text{ “OR” } [m]$
 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: $\text{ACC} \leftarrow \text{ACC} \text{ “OR” } x$
 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: $[m] \leftarrow \text{ACC} \text{ “OR” } [m]$
 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: $\text{Program Counter} \leftarrow \text{Stack}$
 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: $\text{Program Counter} \leftarrow \text{Stack}$
 $\text{ACC} \leftarrow x$
 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： $\text{Program Counter} \leftarrow \text{Stack}$
 $\text{EMI} \leftarrow 1$

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： $[\text{m}].0 \leftarrow [\text{m}].7, [\text{m}].(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$

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： $\text{ACC}.0 \leftarrow [\text{m}].7, \text{ACC}.(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$

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： $[\text{m}].(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$
 $[\text{m}].0 \leftarrow \text{C}$
 $\text{C} \leftarrow [\text{m}].7$

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： $\text{ACC}.(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$
 $\text{ACC}.0 \leftarrow \text{C}$
 $\text{C} \leftarrow [\text{m}].7$

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： $[m].7 \leftarrow [m].0$, $[m].i \leftarrow [m].(i+1)$; (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： $ACC.7 \leftarrow [m].0$, $ACC.i \leftarrow [m].(i+1)$; (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： $[m].i \leftarrow [m].(i+1)$; (i=0~6)

$[m].7 \leftarrow C$

$C \leftarrow [m].0$

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： $ACC.i \leftarrow [m].(i+1)$; (i=0~6)

$ACC.7 \leftarrow C$

$C \leftarrow [m].0$

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： $ACC \leftarrow ACC + [\overline{m}] + C$

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： $[m] \leftarrow \overline{ACC} + [\overline{m}] + C$
 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。
 $ACC \leftarrow ([m]-1)$
 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： $[m] \leftarrow FFH$
 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： $[m].i \leftarrow 1$
 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； $[m] \leftarrow [m]+1$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZE 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； $ACC \leftarrow ([m]+1)$

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 $[m].i \neq 0$ ，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： $ACC \leftarrow ACC + \overline{[m]} + 1$

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： $ACC \leftarrow ACC + \overline{x} + 1$

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： $[m] \leftarrow ACC + [\bar{m}] + 1$
 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： $[m].7 \sim [m].4 \leftrightarrow [m].3 \sim [m].0$
 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： $ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$
 $ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$
 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 $ACC \leftarrow [m]$ 。
 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 and TBLH
 说明： 本指令是将表格指针指向程序寄存器当前页，将低字节送到存储器，高字节直接送到 TBLH 寄存器内。

运算过程： **[m] ← 程序存储器低字节**
TBLH ← 程序存储器高字节

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 and TBLH
 说明： 本指令是将 TABLE 指针指向程序寄存器最后页，将低字节送到存储器，高字节直接送到 TBLH 寄存器内。

运算过程： **[m] ← 程序存储器低字节**
TBLH ← 程序存储器高字节

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
—	—	—	√	—	—

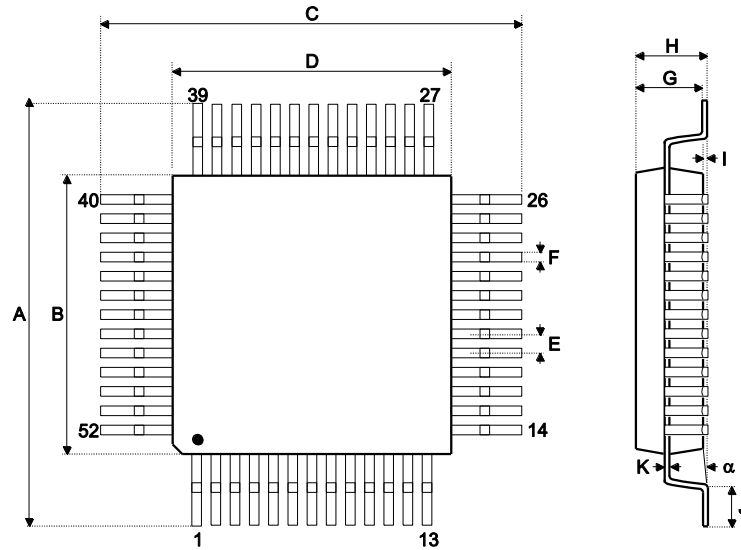
XOR A, x 累加器与数据存储器做“异或”运算，结果放入累加器
 说明： 本指令是把累加器值与立即数做逻辑异或，结果放到累加器。

运算过程： **ACC ← ACC “XOR” x**
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

封装信息

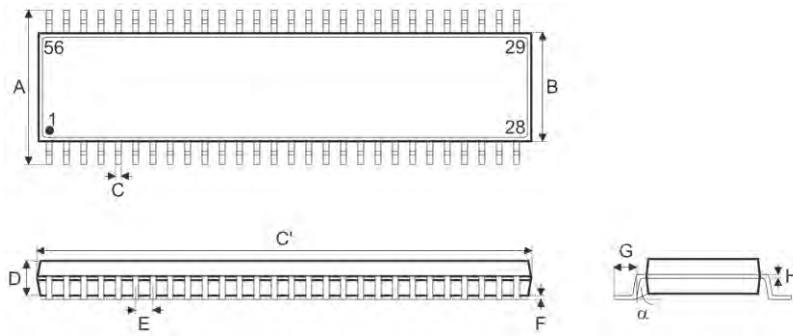
52-pin QFP (14mm×14mm)外形尺寸



符号	尺寸(单位: inch)		
	最小	正常	最大
A	0.681	—	0.689
B	0.547	—	0.555
C	0.681	—	0.689
D	0.547	—	0.555
E	—	0.039	—
F	—	0.016	—
G	0.098	—	0.122
H	—	—	0.134
I	—	0.004	—
J	0.029	—	0.041
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸(单位: mm)		
	最小	正常	最大
A	17.30	—	17.50
B	13.90	—	14.10
C	17.30	—	17.50
D	13.90	—	14.10
E	—	1.00	—
F	—	0.40	—
G	2.50	—	3.10
H	—	—	3.40
I	—	0.10	—
J	0.73	—	1.03
K	0.10	—	0.20
α	0°	—	7°

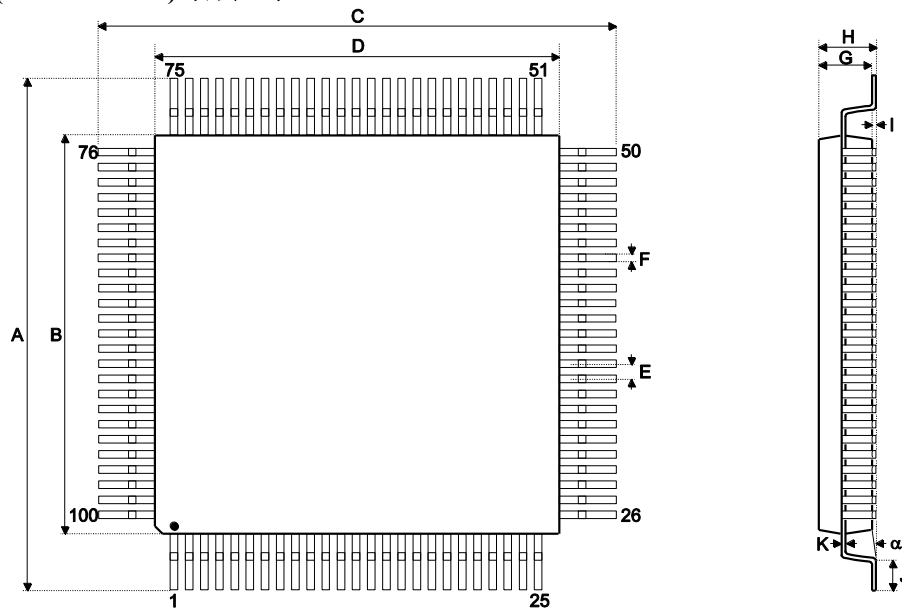
56-pin SSOP (300mil) 外形尺寸



符号	尺寸(单位: inch)		
	最小	正常	最大
A	0.395	—	0.420
B	0.291	—	0.299
C	0.008	—	0.012
C'	0.720	—	0.730
D	0.089	—	0.099
E	—	0.025	—
F	0.004	—	0.010
G	0.025	—	0.035
H	0.004	—	0.012
α	0°	—	8°

符号	尺寸(单位: mm)		
	最小	正常	最大
A	10.03	—	10.67
B	7.39	—	7.59
C	0.20	—	0.30
C'	18.29	—	18.54
D	2.26	—	2.51
E	—	0.64	—
F	0.10	—	0.25
G	0.64	—	0.89
H	0.10	—	0.30
α	0°	—	8°

100-pin LQFP (14mm×14mm)外形尺寸



符号	尺寸(单位: inch)		
	最小	正常	最大
A	0.626	—	0.634
B	0.547	—	0.555
C	0.626	—	0.634
D	0.547	—	0.555
E	—	0.020	—
F	—	0.008	—
G	0.053	—	0.057
H	—	—	0.063
I	—	0.004	—
J	0.018	—	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸(单位: mm)		
	最小	典型	最大
A	15.90	—	16.10
B	13.90	—	14.10
C	15.90	—	16.10
D	13.90	—	14.10
E	—	0.50	—
F	—	0.20	—
G	1.35	—	1.45
H	—	—	1.60
I	—	0.10	—
J	0.45	—	0.75
K	0.10	—	0.20
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（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: 0755-8616-9722

Holtek Semiconductor(USA), 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.holtek.com

Copyright © 2011 by HOLTEK SEMICONDUCTOR INC.

使用指南中所出现的信息在出版当时相信是正确的，然而盛群对于说明书的使用不负任何责任。文中提到的应用目的仅仅是用来做说明，盛群不保证或表示这些没有进一步修改的应用将是适当的，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>