



# JZ8M117

**8位MTP微控制器**

**用户数据手册**



## 声明

本资料内容，随产品的改进，会进行相应更新，恕不另行通知。使用本资料前请咨询我司销售人员，以保证本资料内容为最新版本。

请在本资料所记载的极限范围内使用本产品，因使用不当造成的损失，我司不承担其责任。

尽管本公司一向致力于提高产品质量与可靠性，但是半导体产品本身有一定的概率发生故障或错误工作，为防止因此类事故而造成的人身伤害或财产损失，请在使用过程中充分留心安全设计。

将本产品或者本资料出口海外时，应当遵守适用的进出口管制法律法规。

未经本公司许可，严禁以任何形式复制或转载本资料的部分或全部内容。

本资料测试数据仅供参考，实际数据以目标样机测试为准。



## 目 录

1. 单片机特点.....	7
1.1. 系统特性.....	7
1.2. 系统功能.....	7
1.3. CPU 特点.....	7
1.4. 封装信息.....	7
2. 系统概述和方框图.....	8
3. 引脚分配及功能说明.....	9
4. 中央处理器 (CPU).....	11
4.1. 存储器.....	11
4.1.1. 程序存储器 (ROM).....	11
4.1.2. 数据存储器 (SRAM).....	11
4.1.3. 系统寄存器.....	12
4.1.3.1. 标志寄存器(FLAG), 地址 = 0x00.....	13
4.1.3.2. 杂项寄存器(MISC), 地址 = 0x08.....	13
4.2. 寻址方式.....	13
4.3. 堆栈.....	13
4.3.1. 堆栈指针寄存器(SP), 地址 = 0x02.....	13
4.4. 程序选项 Code Options.....	14
5. 振荡器和系统时钟.....	15
5.1. 内部高频振荡器和内部低频振荡.....	15
5.2. 外部晶体振荡器.....	15
5.2.1. 外部晶体振荡器控制寄存器(EOSCR), 地址 = 0x0A.....	16
5.2.2. 外部晶体振荡器的使用及注意事项.....	16
5.3. 系统时钟与 IHRC 频率校准.....	17
5.3.1. 系统时钟.....	17
5.3.1.1. 时钟控制寄存器(CLKMD), 地址 = 0x03.....	18
5.3.2. 频率校准.....	18
5.3.2.1. 特别声明.....	20
5.3.3. 系统时钟切换.....	20
6. 系统复位与电压检测.....	21
6.1. 上电复位(POR).....	21
6.2. 低电压复位(LVR).....	21
6.3. 看门狗超时溢出复位.....	23
6.4. 外部复位(PRSTB).....	24
6.5. 系统电压检测器 - LVD.....	24
6.5.1. 低电压检测寄存器(LVDC), 地址 = 0x2D.....	24
7. 系统工作模式.....	25
7.1. 省电模式("stopexe").....	25
7.2. 掉电模式("stopsys").....	26
7.3. 唤醒.....	26
8. 中断.....	27
8.1. 中断允许寄存器(INTEN), 地址 = 0x04.....	28
8.2. 中断请求寄存器(INTRQ), 地址 = 0x05.....	29
8.3. 中断边缘选择寄存器(INTEGS), 地址 = 0x0C.....	29



8.4. 中断工作流程.....	30
8.5. 中断的一般步骤.....	30
8.6. 使用中断举例.....	31
<b>9. I/O 端口.....</b>	<b>32</b>
9.1. IO 相关寄存器.....	32
9.1.1. 端口 A 数字输入启用寄存器(PADIER), 地址 = 0x0D.....	32
9.1.2. 端口 B 数字输入启用寄存器(PBDIER), 地址 = 0x0E.....	32
9.1.3. 端口 A 数据寄存器(PA), 地址 = 0x10.....	32
9.1.4. 端口 A 控制寄存器(PAC), 地址 = 0x11.....	32
9.1.5. 端口 A 上拉控制寄存器(PAPH), 地址 = 0x12.....	32
9.1.6. 端口 A 下拉控制寄存器(PAPL), 地址 = 0x13.....	33
9.1.7. 端口 B 数据寄存器(PB), 地址 = 0x14.....	33
9.1.8. 端口 B 控制寄存器(PBC), 地址 = 0x15.....	33
9.1.9. 端口 B 上拉控制寄存器(PBPH), 地址 = 0x16.....	33
9.1.10. 端口 B 下拉控制寄存器(PBPL), 地址 = 0x17.....	33
9.2. IO 结构及功能.....	34
9.2.1. IO 引脚的结构.....	34
9.2.2. IO 引脚的一般功能.....	34
9.2.3. IO 使用与设定.....	35
<b>10. Timer / PWM 计数器.....</b>	<b>36</b>
10.1. 16 位计数器 (Timer16).....	36
10.1.1. Timer16 介绍.....	36
10.1.2. Timer16 控制寄存器(T16M), 地址 = 0x06.....	37
10.1.3. Timer16 溢出时间.....	38
10.2. 8 位 PWM 计数器(Timer2,Timer3).....	38
10.2.1. Timer2、Timer3 相关寄存器.....	39
10.2.1.1. Timer2 分频寄存器(TM2S), 地址 = 0x1E.....	39
10.2.1.2. Timer2 控制寄存器(TM2C), 地址 = 0x1C.....	40
10.2.1.3. Timer2 计数寄存器(TM2CT), 地址 = 0x1D.....	40
10.2.1.4. Timer2 上限寄存器(TM2B), 地址 = 0x1F.....	40
10.2.1.5. Timer3 计数寄存器(TM3CT), 地址 = 0x33.....	40
10.2.1.6. Timer3 分频寄存器(TM3S), 地址 = 0x34.....	41
10.2.1.7. Timer3 上限寄存器(TM3B), 地址 = 0x35.....	41
10.2.1.8. Timer3 控制寄存器(TM3C), 地址 = 0x32.....	41
10.2.2. 使用 Timer2 产生定期波形.....	42
10.2.3. 使用 Timer2 产生 8 位 PWM 波形.....	43
10.2.4. 使用 Timer2 产生 6 位 PWM 波形.....	44
<b>11. 特殊功能.....</b>	<b>45</b>
11.1. 比较器.....	45
11.1.1. 比较器控制寄存器(GPCC), 地址 = 0x18.....	46
11.1.2. 比较器选择寄存器(GPCS), 地址 = 0x19.....	46
11.1.3. 内部参考电压 (V <sub>internal R</sub> ).....	47
11.1.4. 使用比较器.....	49
11.1.5. 使用比较器和 Bandgap 参考电压生成器.....	50
11.2. 触摸功能.....	51
11.2.1. 触摸相关寄存器.....	53
11.2.1.1. 触摸选项寄存器(TS), 地址 = 0x20.....	53
11.2.1.2. 触摸充电控制寄存器(TCC), 地址 = 0x21.....	53



11.2.1.3. 触摸按键使能 2 寄存器 (TKE2), 地址 = 0x22.....	54
11.2.1.4. 触摸按键使能 1 寄存器 (TKE1), 地址 = 0x24.....	54
11.2.1.5. 触摸按键充电计数高位寄存器(TKCH), 地址= 0x2B.....	54
11.2.1.6. 触摸按键充电计数低位寄存器(TKCL), 地址= 0x2C.....	54
<b>12. 仿真注意事项.....</b>	<b>55</b>
<b>13. 烧录方法.....</b>	<b>56</b>
13.1. 普通烧录模式.....	56
13.2. 限压烧录模式.....	56
13.3. 在板烧录 (On-Board Writing) .....	57
<b>14. 直流交流电气特性.....</b>	<b>58</b>
14.1. 绝对最大值.....	58
14.2. 器件电气特性.....	58
14.3. IHRC 频率与 VDD 关系曲线图 (校准到 16MHz) .....	60
14.4. ILRC 频率与 VDD 关系曲线图.....	60
14.5. IHRC 频率与温度关系曲线图 (校准到 16MHz) .....	61
14.6. ILRC 频率与温度关系曲线图.....	61
14.7. 工作电流与 VDD、系统时钟 CLK = IHRC/n 关系曲线图.....	62
14.8. 工作电流与 VDD、系统时钟 CLK = ILRC/n 关系曲线图.....	62
14.9. 工作电流与 VDD、系统时钟 CLK = 32KHz EOSC / n 关系曲线图.....	63
14.10. 工作电流与 VDD、系统时钟 CLK = 1MHz EOSC / n 关系曲线图.....	63
14.11. 工作电流与 VDD、系统时钟 CLK = 4MHz EOSC / n 关系曲线图.....	64
14.12. IO 引脚上拉阻抗曲线图.....	64
14.13. IO 引脚下拉阻抗曲线图.....	65
14.14. IO 引脚输出驱动电流( $I_{OH}$ ) 和灌电流 ( $I_{OL}$ )曲线图.....	65
14.15. IO 引脚输入高/低阈值电压( $V_{IH}/V_{IL}$ )曲线图.....	66
14.16. 掉电模式消耗电流( $I_{PD}$ )与省电模式消耗电流( $I_{PS}$ )曲线图.....	66
<b>15. 指令.....</b>	<b>67</b>
15.1. 指令表.....	68



### 修订历史:

修 订	日 期	描 述
0.00	2019/08/23	初版



## 1. 单片机特点

### 1.1. 系统特性

高抗干扰 (High EFT) 系列

特别适用于 AC 电源供电的、阻容降压电路的、需要较强抗干扰能力的, 或有高 EFT 安规测试要求 ( $\pm 4\text{KV}$ ) 的产品

- ◆ 工作温度范围:  $-40^{\circ}\text{C} \sim 85^{\circ}\text{C}$

ESD > 8 KV

产品不良率保证低于 500 PPM。

- ◆ 产品寿命保证在 5 年以上 (焊接温度 $\leq 260^{\circ}\text{C}$ )。

### 1.2. 系统功能

2KW MTP 程序空间 (可编程 1000 次)

128 Bytes 数据空间

最大 7 IO 引脚可选择为触摸按键

一个硬件 16 位定时器

两个 8 位定时器可产生 6/7/8-bit PWM 波形

一个硬件比较器

最大 8 IO 引脚带可选择的上拉/下拉电阻

每个 IO 引脚都可设定为唤醒功能

Bandgap 电路提供 1.20V 参考电压

时钟源: IHRC、ILRC 和 EOSC (XTAL)

8 级可选择的 LVR (Low Voltage Reset) 复位电压从 1.8V 到 4.5V

16 级可选择的 LVD (Low Voltage Detect) 检测电压从 1.8V 到 4.5V

三个可选择的外部中断引脚

### 1.3. CPU 特点

8 位高性能精简指令集 CPU

86 条高效的指令

绝大部分指令都是单周期(1T)指令

可程序设定的堆栈指针和堆栈深度

数据存取支持直接和间接寻址模式

用数据存储寄存器即可当作间接寻址模式的数据指针(index pointer)

寄存器地址空间、数据存储空间、MTP 程序空间三者互相独立

### 1.4. 封装信息

JZ8M117-S08A: SOP8A  
(150mil)

JZ8M117-S08B: SOP8B  
(150mil)



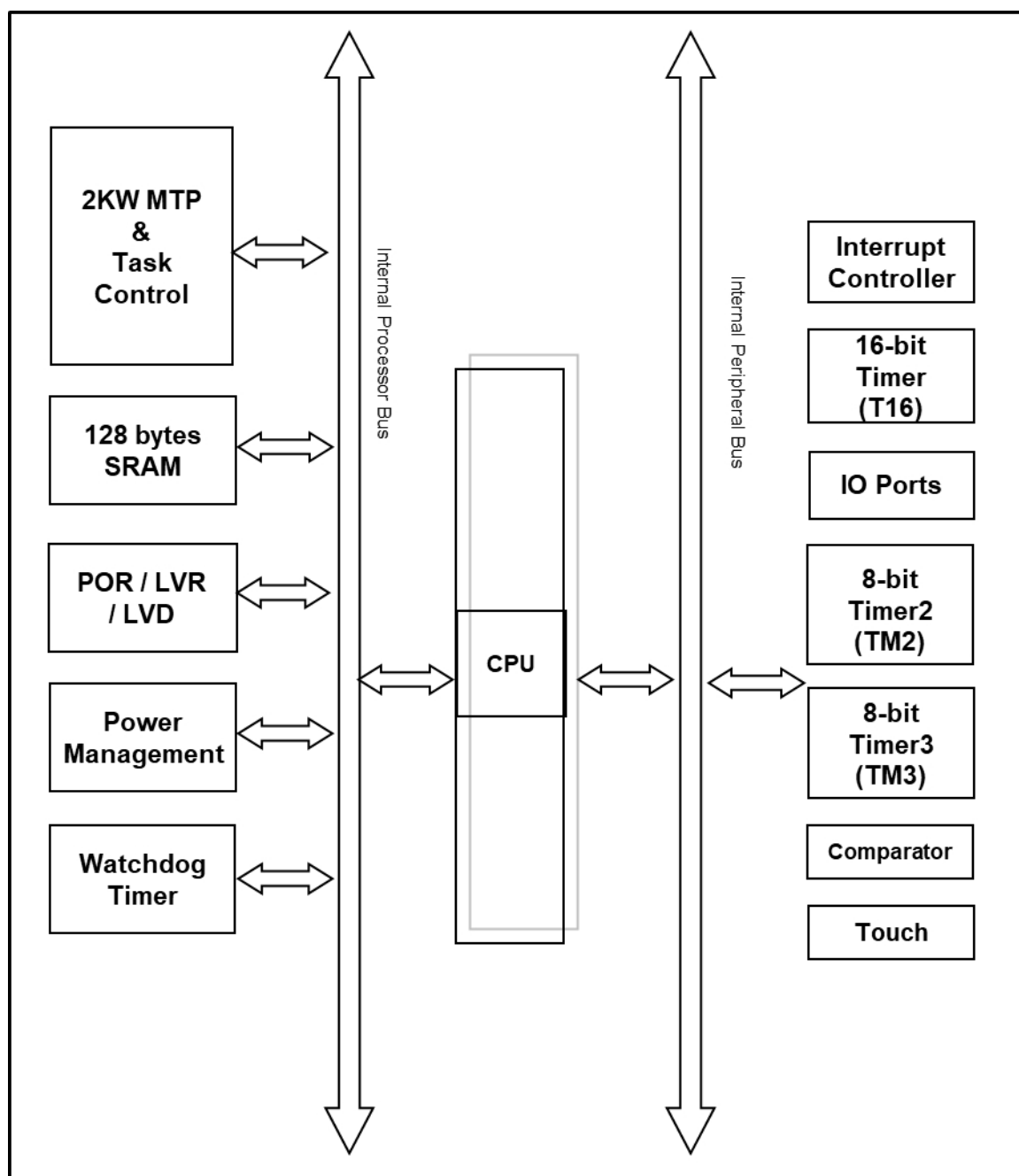
## 2. 系统概述和方框图

JZ8M117 是一个 IO 类型，静态 MTP 单片机。它运用 RISC 的架构基础使大部分指令的执行周期都是单周期，只有少部分间接寻址的指令需要两个指令周期。

JZ8M117 内置 2KW MTP 程序存储器以及 128 字节数据存储器。

JZ8M117 内置最多 7 个按键触摸控制器以及一个硬件比较器。

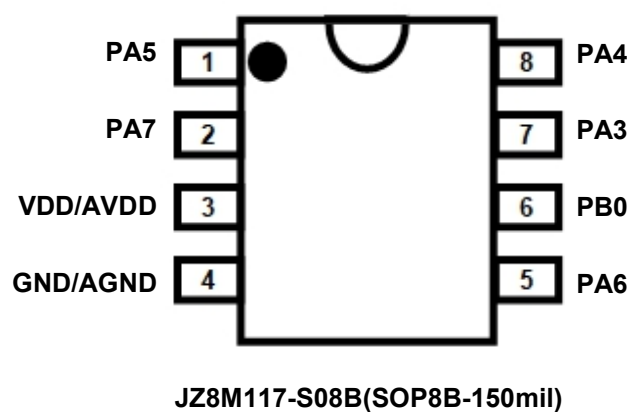
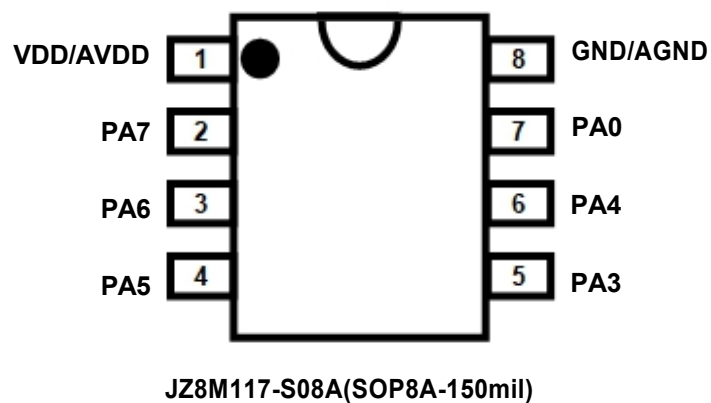
JZ8M117 提供一个 16 位的硬件计数器(Timer16)、两个带 PWM 的 8 位计数器(Timer2、Timer3)。







### 3. 引脚分配及功能说明





# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

引脚名称	输入/输出				特殊功能							
	I/O	上拉	下拉	唤醒	晶振	比较器	PWM	触摸	CS 电容	外部中断	外部复位	烧录
PA0	√	√	√	√		CO	TM2PWM	TK7		INT0		
PA3	√	√	√	√		CIN-	TM2PWM	TK5				√
PA4	√	√	√	√		CIN+ CIN-	TM3PWM	TK6				
PA5	√	√	√	√			TM3PWM	TK9		INT0A	√	√
PA6	√	√	√	√	√			TK8				√
PA7	√	√	√	√	√			TK10	√			
PB0	√	√	√	√			TM2PWM	TK11		INT1		
PB7	√	√	√	√		CIN-	TM3PWM	TK4	√			
VDD / AVDD												√
GND / AGND												√
注意	1. 所有 I/O 引脚都具有：施密特触发器输入；CMOS 电压基准位。 2. 当某引脚作为 PWM 输出端口时，其 IO 功能自动停用。 3. 当 PA5 引脚设定成输入时，对于需要高抗干扰能力的系统，请串接 33Ω 电阻。 4. VDD 是 IC 电源，AVDD 为模拟正电源。在 IC 内部，AVDD 与 VDD 连在一起(double bonding)，而外部为相同引脚。 5. GND 是 IC 接地引脚，而 AGND 是模拟接地引脚。在 IC 内部，AGND 与 GND 连在一起(double bonding)，而外部为相同引脚。											



## 4. 中央处理器 (CPU)

### 4.1. 存储器

#### 4.1.1. 程序存储器 (ROM)

JZ8M117 的程序存储器记忆体是 MTP（可多次编程），用来存放数据（包含：数据、表格和中断入口）和要执行的程序指令。JZ8M117 的程序存储器容量为 2KW，如表 1 所示。

复位之后，程序从 0x000 开始（通常是 *goto* FPPA0），中断入口在 0x010。

MTP 存储器从地址“0x7E0 to 0x7FF”供系统使用，从“0x001 ~ 0x00F”和“0x011~0x7DF”地址空间是用户的程序空间。

MTP 程序存储器最后 32 个地址空间是被保留给系统使用，如：校验码，序列号等。

地址	功能
0x000	FPP0 复位 – <i>goto</i> 指令
0x001	用户程序区
•	•
0x00F	用户程序区
0x010	中断入口地址
0x011	用户程序区
•	•
0x7DF	用户程序区
0x7E0	系统使用
•	•
0x7FF	系统使用

表 1: JZ8M117 程序存储器结构

#### 4.1.2 数据存储器 (SRAM)

JZ8M117 的数据存储器共有 128 字节，数据存取可以是字节或位的操作。除了存储数据外，数据存储器还可以担任间接存取方式的数据指针，以及堆栈存储器。



# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

### 4.1.3. 系统寄存器

JZ8M117 的寄存器地址空间与数据存储空间、MTP 程序空间三者互相独立。以下是 JZ8M117 的各寄存器存放地址及简要描述：

	+0	+1	+2	+3	+4	+5	+6	+7
0x00	<i>FLAG</i>	-	<i>SP</i>	<i>CLKMD</i>	<i>INTEN</i>	<i>INTRQ</i>	<i>T16M</i>	-
0x08	<i>MISC</i>	-	<i>EOSCR</i>	-	<i>INTEGS</i>	<i>PADIER</i>	<i>PBDIER</i>	-
0x10	<i>PA</i>	<i>PAC</i>	<i>PAPH</i>	<i>PAPL</i>	<i>PB</i>	<i>PBC</i>	<i>PBPH</i>	<i>PBPL</i>
0x18	<i>GPCC</i>	<i>GPCS</i>	-	-	<i>TM2C</i>	<i>TM2CT</i>	<i>TM2S</i>	<i>TM2B</i>
0x20	<i>TS</i>	<i>TCC</i>	<i>TKE2</i>	-	<i>TKE1</i>	-	-	-
0x28	<i>TPS2</i>	-	-	<i>TKCH</i>	<i>TKCL</i>	<i>LVDC</i>	-	-
0x30	-	-	<i>TM3C</i>	<i>TM3CT</i>	<i>TM3S</i>	<i>TM3B</i>	-	-

*FLAG*: 标志寄存器

*SP*: 堆栈指针寄存器

*CLKMD*: 时钟控制寄存器

*EOSCR*: 外部晶体振荡器控制寄存器

*INTEN*: 中断允许寄存器

*INTRQ*: 中断请求寄存器

*INTEGS*: 中断选择寄存器

*MISC*: 杂项寄存器

*PA*: 端口 A 数据寄存器

*PAC*: 端口 A 控制寄存器

*PAPH*: 端口 A 上拉控制寄存器

*PAPL*: 端口 A 下拉控制寄存器

*PADIER*: 端口 A 数字输入启用寄存器

*PB*: 端口 B 数据寄存器

*PBC*: 端口 B 控制寄存器

*PBPH*: 端口 B 上拉控制寄存器

*PBPL*: 端口 B 下拉控制寄存器

*PBDIER*: 端口 B 数字输入启用寄存器

*GPCC*: 比较器控制寄存器

*GPCS*: 比较器选择寄存器

*T16M*: Timer16 控制寄存器

*TM2C* / *TM3C*: Timer2 / Timer3 控制寄存器

*TM2CT* / *TM3CT*: Timer2 / Timer3 计数寄存器

*TM2S* / *TM3S*: Timer2 / Timer3 分频寄存器

*TM2B* / *TM3B*: Timer2 / Timer3 上限寄存器

*TS*: 触摸选项寄存器

*TCC*: 触摸充电控制寄存器

*TKE1* / *TKE2*: 触摸按键使能寄存器

*TKCH* / *TKCL*: 触摸按键充电计数高/低位寄存器

*LVDC*: 低电压检测寄存器

*TPS2*: 触摸参数设置寄存器



### 4.1.3.1. 标志寄存器(FLAG), 地址 = 0x00

位	初始值	读/写	描述
7 - 4	-	-	保留。这 4 个位读值为“1”。
3	-	读/写	OV（溢出标志）。当数学运算溢出时，这一位会设置为 1。
2	-	读/写	AC（辅助进位标志）。两个条件下，此位设置为 1： (1)是进行低半字节加法运算产生进位 (2)减法运算时，低半字节向高半字节借位。
1	-	读/写	C（进位标志）。有两个条件下，此位设置为 1：(1)加法运算产生进位 (2)减法运算有借位。进位标志还受带进位标志的 shift 指令影响。
0	-	读/写	Z（零）。此位将被设置为 1，当算术或逻辑运算的结果是 0；否则将被清零。

### 4.1.3.2. 杂项寄存器(MISC), 地址 = 0x08

位	初始值	读/写	描述
7 - 6	-	-	保留。
5	0	只写	唤醒时间 0 / 1 : 3000 ILRC (Slow) / 45 ILRC (Fast)
4 - 3	-	-	保留。
2	0	只写	停用 LVR 功能： 0 / 1: 启用 / 停用
1 - 0	00	只写	看门狗时钟超时时间设定： 00: 8K 个 ILRC 时钟周期 01: 16K 个 ILRC 时钟周期 10: 64K 个 ILRC 时钟周期 11: 256K 个 ILRC 时钟周期

## 4.2. 寻址方式

数据存储器的间接存取方式，是以数据存储器当作数据指针来存取数据字节。所有的数据存储器，都可以拿来当作数据指针，这可以让单片机的资源利用率最大化。JZ8M117 的数据存储器 128 字节全部都可以用间接方式来存取。

位寻址只能定义在 RAM 区的 0x00 到 0x3F 空间。

## 4.3. 堆栈

堆栈存储器是定义在数据存储器里。堆栈存储器的堆栈指针是定义在堆栈指针寄存器；堆栈存储器深度是由使用者定义的。用户可以依其程序需求来订定所需要堆栈存储器的大小，以保持最大的弹性。

### 4.3.1. 堆栈指针寄存器(SP), 地址 = 0x02

位	初始值	读/写	描述
7 - 0	-	读/写	堆栈指针寄存器。读出当前堆栈指针，或写入以改变堆栈指针。请注意 0 位必须维持为 0。因程序计数器是 16 位。



### 4.4. 程序选项 Code Options

选项	选择	描述
Security	<b>Enable(默认)</b>	MTP 内容加密 7/8 words
	Disable	MTP 内容不加密，程序可被读回
LVR	4.0V	选择 LVR = 4.0V
	3.5V	选择 LVR = 3.5V
	3.0V	选择 LVR = 3.0V
	2.7V	选择 LVR = 2.7V
	<b>2.5V(默认)</b>	选择 LVR = 2.5V
	2.2V	选择 LVR = 2.2V
	2.0V	选择 LVR = 2.0V
	1.8V	选择 LVR = 1.8V
PA3_PA4_Drive	Strong	PA3 & PA4 的驱动电流/灌电流为 strong
	<b>Normal(默认)</b>	PA3 & PA4 的驱动电流/灌电流为 normal
TMx_Source	<b>16MHZ(默认)</b>	当 TM2C[7:4]= 0010, TM2 时钟源= IHRC = 16MHZ 当 TM3C[7:4]= 0010, TM3 时钟源= IHRC = 16MHZ
	32MHZ	当 TM2C[7:4]= 0010, TM2 时钟源 = IHRC*2 = 32MHZ 当 TM3C[7:4]= 0010, TM3 时钟源 = IHRC*2 = 32MHZ (仿真器不支持)
TMx_Bit	<b>6 Bit(默认)</b>	当 TM2S.7=1, TM2 是 6 位 PWM 当 TM3S.7=1, TM3 是 6 位 PWM
	7 Bit	当 TM2S.7=1, TM2 是 7 位 PWM 当 TM3S.7=1, TM3 是 7 位 PWM (仿真器不支持)
Comparator Edge	<b>All Edge(默认)</b>	上升缘和下降缘都触发中断
	Rising Edge	仅上升缘触发中断
	Falling Edge	仅下降缘触发中断
GPC_PWM	<b>Disable(默认)</b>	比较器和 PWM 相互独立
	Enable	比较器输出控制 PWM 输出 (仿真器不支持)
Interrupt Src0	<b>PA.0(默认)</b>	选择 INTEN/INTRQ.Bit0 为 PA.0
	PA.5	选择 INTEN/INTRQ.Bit0 为 PA.5 (仿真器不支持)
CS_Sel	<b>PA7(默认)</b>	配置引脚 PA7/CS 为触摸的 CS 脚
	PB7	配置引脚 PB7/CS 为触摸的 CS 脚
	Disable	配置引脚 PA7/PB7/CS 为正常的 IO 脚
EMI	Disable	停用 EMI 优选项
	<b>Enable(默认)</b>	系统时钟会轻微调整以获得更好的 EMI 性能



### 5. 振荡器和系统时钟

JZ8M117 提供 3 个振荡器电路：外部晶体振荡器(EOSC)、内部高频 RC 振荡器(IHRC)、内部低频 RC 振荡器(ILRC)。

这 3 个振荡器可以分别用寄存器 *EOSCR.7*, *CLKMD.4* 与 *CLKMD.2* 启用或停用，使用者可以选择这 3 个振荡器之一作为系统时钟源，并透过 *CLKMD* 寄存器来改变系统时钟频率，以满足不同的系统应用。

振荡器硬件	启用或停用选择
EOSC	<i>EOSCR.7</i>
IHRC	<i>CLKMD.4</i>
ILRC	<i>CLKMD.2</i>

表 2: JZ8M117 提供 3 个振荡器电路

#### 5.1. 内部高频振荡器和内部低频振荡

IHRC、ILRC 的频率会因工厂生产、电源电压和温度的变化而变化，请参阅 IHRC、ILRC 频率和 VDD、温度的测量图表。

JZ8M117 烧录工具提供 IHRC 频率校准（通常校准到 16MHz）功能，以此来消除工厂生产引起的频率漂移。ILRC 没有校准操作，对于需要精准定时的应用请不要使用 ILRC 的时钟当作参考时间。

#### 5.2. 外部晶体振荡器

外部晶体振荡器的工作频率范围可以从 32KHz 至 4MHz，JZ8M117 不支持频率在 4MHz 以上的振荡器。图 1 显示了使用外部晶体振荡器的硬件连接。

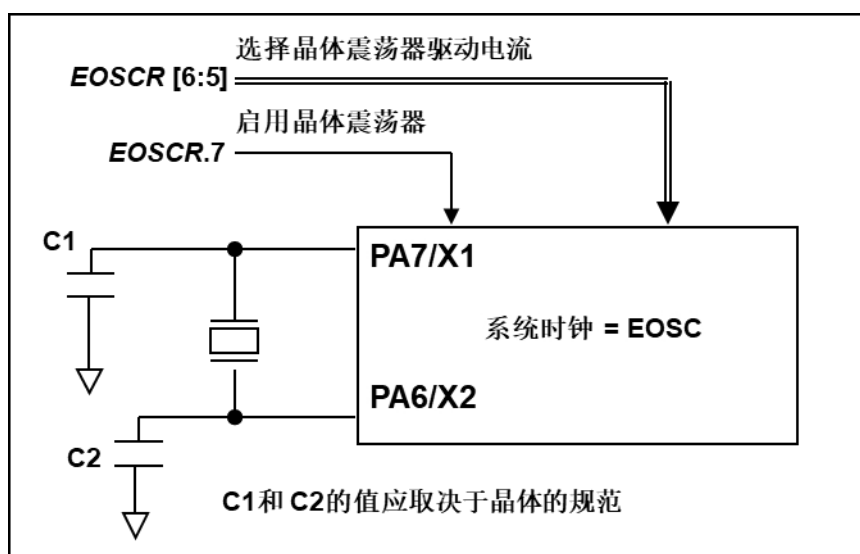


图 1: 外部晶体振荡器的硬件连接

**521. 外部晶体振荡器控制寄存器(EOSCR), 地址 = 0x0A**

位	初始值	读/写	描 述
7	0	只写	使能外部晶体振荡器。0 / 1: 停用/使能
6 – 5	00	只写	晶体振荡器的选择。 00: 保留 01: 低驱动电流。适用于较低频率晶体, 例如: 32KHz 10: 中驱动电流。适用于中等频率晶体, 例如: 1MHz 11: 高驱动电流。适用于较高频率晶体, 例如: 4MHz
4 – 0	-	-	保留。请设为 0。

**522 外部晶体振荡器的使用及注意事项**

除了晶振的选择外, 外部电容器和寄存器 *EOSCR* 相关选项也应该适度调整以求得有良好的正弦波。*EOSCR.7* 是用开启晶体振荡器硬件模块, *EOSCR.6* 和 *EOSCR.5* 用于设置振荡器不同的驱动电流, 以满足晶体振荡器不同频率的要求。

表 3 显示了不同的晶体振荡器 *C1* 和 *C2* 的推荐值, 同时也显示其对应条件下测量的起振时间。由于晶体或谐振器有其自身的特点, 不同类型的晶体或谐振器的启动时间可能会略有不同, 请参考其规格并选择恰当的 *C1* 和 *C2* 电容值。

频率	C1	C2	起振时间	条件
4MHz	4.7pF	4.7pF	6ms	( <i>EOSCR</i> [6:5]=11)
1MHz	10pF	10pF	11ms	( <i>EOSCR</i> [6:5]=10)
32KHz	22pF	22pF	450ms	( <i>EOSCR</i> [6:5]=01)

表 3: 晶体振荡器 *C1* 和 *C2* 推荐值

使用晶振时 *PA7* 和 *PA6* 的配置:

- (1) *PA7* 和 *PA6* 设定为输入;
- (2) *PA7* 和 *PA6* 内部上拉电阻设为关闭;
- (3) 用 *PADIER* 寄存器将 *PA6* 和 *PA7* 设为模拟输入, 防止漏电。

**注意:** 请务必仔细阅读《PMC-APN013》之内容, 并据此合理使用晶体振荡器。如因用户的晶体振荡器的质量不佳、使用条件不合理、PCB 清洁剂残留漏电、或是 PCB 板布局不合理等等用户原因, 造成的慢起振或不起振情况, 我司不对此负责。

使用晶体振荡器时, 使用者必须特别注意振荡器的稳定时间。稳定时间将取决于振荡器频率、晶型、外部电容和电源电压。在系统时钟切换到晶体振荡器之前, 使用者必须确保晶体振荡器是稳定的, 相关参考程序如下所示:





```
void    FPPA0 (void)
{
    .ADJUST_IC  SYSCLK=IHRC/16, IHRC=16MHz, VDD=5V
    ...
    $    EOSCR  Enable, 4Mhz;      // EOSCR = 0b110_00000;
    $    T16M    EOSC, /1, BIT13;  // T16M.Bit13 由 0->1 時, INTRQ.T16 => 1
                                      //假设此时晶体振荡器已稳定

    WORD    count    =    0;
    stt16    count;
    Intrq.T16 =    0;
    while (! Intrq.T16)  NULL;      // 从 0x0000 算到 0x2000, 然后设置 INTRQ.T16
    clkmd=0xB4;                    // 切换系统时钟到 EOSC;
    clkmd.4 = 0;                    // 关闭 IHRC
    ...
}
```

需要注意，在进入掉电模式前，为保证系统不会被误唤醒，要确保外部晶体振荡器已完全关闭。

### 5.3.系统时钟与IHRC 频率校准

#### 5.3.1. 系统时钟

系统时钟的时钟源有 EOSC，IHRC 和 ILRC，JZ8M117 的时钟系统的硬件框图如图 2 所示。

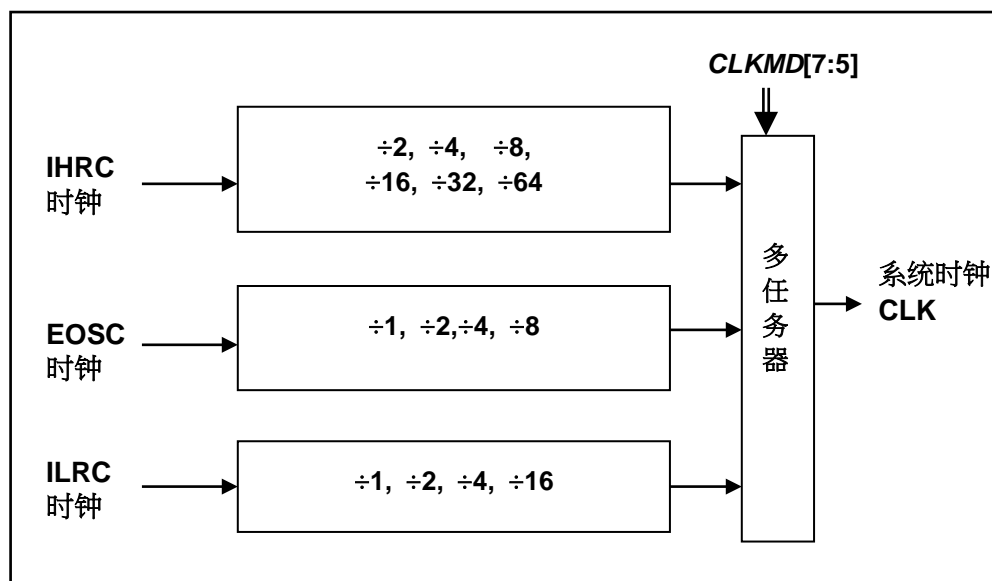


图 2：系统时钟源选择



### 5.3.1.1. 时钟控制寄存器(CLKMD), 地址 = 0x03

位	初始值	读/写	描 述
7 – 5	111	读/写	系统时钟选择
			类型 0, CLKMD[3]=0
			类型 1, CLKMD[3]=1
7 – 5	111	读/写	000: IHRC/4
			001: IHRC/2
			010: 保留
			011: EOSC/4
			100: EOSC/2
			101: EOSC
			110: ILRC/4
			111: ILRC (默认)
			其他: 保留
4	1	读/写	内部高频 RC 振荡器功能。 0/1: 停用/启用
3	0	读/写	时钟类型选择。这个位是用来选择位 7~位 5 的时钟类型。 0 / 1: 类型 0 / 类型 1
2	1	读/写	内部低频 RC 振荡器功能。 0/1: 停用/启用 当内部低频 RC 振荡器功能停用时, 看门狗功能同时被关闭。
1	1	读/写	看门狗功能。 0/1: 停用/启用
0	0	读/写	引脚 PA5/PRSTB 功能。 0 / 1: PA5 / PRSTB

## 5.3.2 频率校准

IHRC 校准的功能是在用户编译程序时序做选择, 校准命令以及校准选项将自动插入到用户程序中。校准命令如下所示:

```
.ADJUST_IC SYSCLK=IHRC/(p1), IHRC=(p2)MHz, VDD=(p3)V;
```

**p1** =2, 4, 8, 16, 32; 以提供不同的系统时钟。

**p2** =16~18; 校准芯片到不同的频率, 通常选择 16MHz。

**p3** =2.2~5.5; 根据不同的电源电压校准芯片。

通常情况下, ADJUST\_IC 是开机后的第一个命令, 用以设定系统的工作频率。IHRC 频率校准的程序只在将程序代码写入 MTP 存储器的时候执行一次, 此后不会再被执行。

如果 IHRC 校准选择不同的选项, 开机后的系统状态也是不同的。IHRC 频率校准以及系统时钟的选项, 如表 4 所示:



# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

SYSCLK	CLKMD	IHRCR	描述
○ Set IHRC / 2	= 34h (IHRC / 2)	有校准	IHRC 校准到 16MHz, CLK=8MHz (IHRC/2)
○ Set IHRC / 4	= 14h (IHRC / 4)	有校准	IHRC 校准到 16MHz, CLK=4MHz (IHRC/4)
○ Set IHRC / 8	= 3Ch (IHRC / 8)	有校准	IHRC 校准到 16MHz, CLK=2MHz (IHRC/8)
○ Set IHRC / 16	= 1Ch (IHRC / 16)	有校准	IHRC 校准到 16MHz, CLK=1MHz (IHRC/16)
○ Set IHRC / 32	= 7Ch (IHRC / 32)	有校准	IHRC 校准到 16MHz, CLK=0.5MHz (IHRC/32)
○ Set ILRC	= E4h (ILRC / 1)	有校准	IHRC 校准到 16MHz, CLK=ILRC
○ Disable	没改变	没改变	IHRC 不校准, CLK 没改变

表 4: IHRC 频率校准选项

下面显示在不同的选项下, JZ8M117 不同的状态:

### **(1).ADJUST\_IC SYSCLK=IHRC/2, IHRC=16MHz, V<sub>DD</sub>=5V**

开机后, CLKMD = 0x34:

- IHRC 的校准频率为 16MHz@V<sub>DD</sub>=5V, 启用 IHRC 的硬件模块
- 系统时钟 = IHRC/2 = 8MHz
- 看门狗被停用, 启用 ILRC, PA5 是在输入模式

### **(2).ADJUST\_IC SYSCLK=IHRC/8, IHRC=16MHz, V<sub>DD</sub>=2.5V**

开机后, CLKMD = 0x3C:

- IHRC 的校准频率为 16MHz@V<sub>DD</sub>=2.5V, 启用 IHRC 的硬件模块
- 系统时钟 = IHRC/8 = 2MHz
- 看门狗被停用, 启用 ILRC, PA5 是在输入模式

### **(3).ADJUST\_IC SYSCLK=ILRC, IHRC=16MHz, V<sub>DD</sub>=5V**

开机后, CLKMD = 0xE4:

- IHRC 的校准频率为 16MHz@V<sub>DD</sub>=5V, 停用 IHRC 的硬件模块
- 系统时钟 = ILRC
- 看门狗被停用, 启用 ILRC, PA5 是在输入模式

### **(4).ADJUST\_IC DISABLE**

开机后, CLKMD 寄存器没有改变 (没有任何动作):

- IHRC 不校准并且 IHRC 模块停用
- 系统时钟 = ILRC 或 IHRC/64
- 看门狗被启用, 启用 ILRC, PA5 是在输入模式



### 5.3.2.1. 特别声明

- (1) IHRC 的校正操作是在 IC 烧录时进行的。
- (2) IC 塑封材料（不论是封装用还是 COB 用的黑胶）的特性会对 IHRC 的频率有一定影响。如果用户在 IC 盖上塑封材料前进行烧录，然后再封上塑封材料，则可能造成 IHRC 的特性偏移超出规格的现象，正常情况下频率会变慢一些。
- (3) 上述问题通常发生在用户使用 COB 封装或是委托我司进行晶圆代烧(QTP)时。此情况下我司将不对频率超出规格的情况负责。
- (4) 用户可按自身经验进行一些补偿性调整，例如把 IHRC 的目标频率调高 0.5%-1%左右，令封装后 IC 的 IHRC 频率更接近目标值。

### 5.3.3. 系统时钟切换

IHRC 校准后，透过 **CLKMD** 寄存器的设定，JZ8M117 系统时钟可以随意在 IHRC，ILRC 和 EOSC 之间切换。但必须注意，不可在切换系统时钟的同时把原时钟源关闭。例如：从 A 时钟源切换到 B 时钟源时，应该先把系统时钟源切换到B，然后再关闭A时钟源。请参阅IDE：“使用手册”->“IC 介绍”->“缓存器介绍”->“CLKMD”。

例 1: 系统时钟从 ILRC 切换到 IHRC/2

```
... // 系统时钟为 ILRC
CLKMD.4 = 1; // 先打开 IHRC，可以提高抗干扰能力
CLKMD = 0x34; // 切换为 IHRC/2，ILRC 不能在这里停用
// CLKMD.2 = 0; // 假如需要，ILRC 可以在这里停用
```

例 2: 系统时钟从 IHRC/2 切换到 EOSC

```
... // 系统时钟为 IHRC/2
CLKMD = 0xB0; // 切换为 EOSC，IHRC 不能在这里停用
CLKMD.4 = 0; // IHRC 可以在这里停用
```

例 3: 系统时钟从 IHRC/2 切换到 IHRC/4

```
... // 系统时钟为 IHRC/2, ILRC 为启用
CLKMD = 0X14; // 切换为 IHRC/4
```

例 4: 系统可能当机，如果同时切换时钟和关闭原来的振荡器

```
... // 系统时钟为 ILRC
CLKMD = 0x30; // 不能从 ILRC 切换到 IHRC/2，同时又关闭 ILRC 振荡器
...
```



## 6. 系统复位与电压检测

引起 JZ8M117 复位的原因有四种：上电复位、LVR 复位、看门狗超时溢出复位和 PRSTB 引脚复位。发生复位后，系统会重新启动，程序计数器会跳跃到地址 0x000，JZ8M117 的所有寄存器将被设置为默认值。

### 6.1. 上电复位(POR)

开机时，POR(Power On Reset)是用于复位 JZ8M117，其时序图如图 3 所示。用户必须确保上电后电源电压稳定。

发生上电复位时，JZ8M117 数据存储器的值处于不确定的状态。

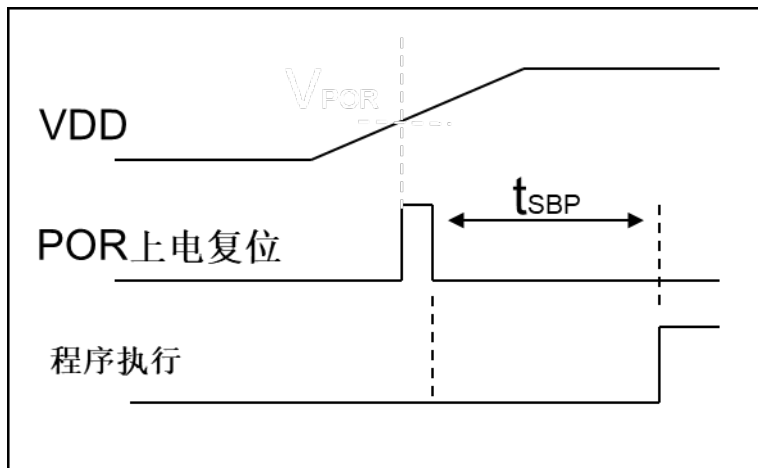


图 3：上电复位时序图

### 6.2. 低电压复位(LVR)

若 VDD 下降低于 LVR(Low Voltage Reset)电压水平，系统会发生 LVR 复位，其时序图如图 4。当 LVR 复位时，JZ8M117 数据存储器的值处于不确定的状态。

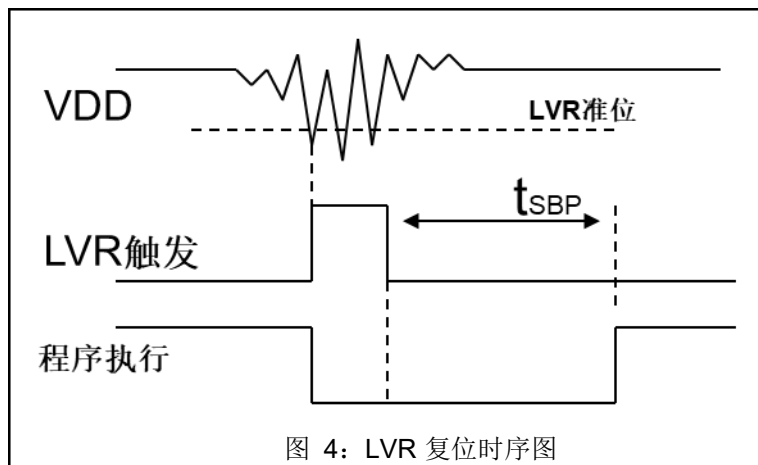


图 4：LVR 复位时序图



LVR 水平的选择在程序编译时进行。使用者必须结合单片机工作频率和电源电压来选择 LVR，才能让单片机稳定工作。下面是工作频率、电源电压和 LVR 水平设定的建议：

系统时钟	VDD	LVR
8MHz	$\geq 3.5V$	$\geq 3.5V$
4MHz	$\geq 2.5V$	$\geq 2.5V$
2MHz	$\geq 2.2V$	$\geq 2.2V$

表 5: LVR 设置参考，与系统频率、VDD 之间的关系

- (1) 只有当 IC 正常起动后，设定 LVR（1.8V ~ 4.0V）才会有效。
- (2) 可以设定寄存器 MISC.2 为 1 将 LVR 关闭，但此时应确保  $V_{DD}$  在最低工作电压以上，否则 IC 可能工作不正常。
- (3) 在省电模式 stopexe 和掉电模式 stopsys 下，LVR 功能无效。

杂项寄存器(MISC)，地址 = 0x08			
位	初始值	读/写	描述
7 - 6	-	-	保留。
5	0	只写	唤醒时间。
4 - 3	-	-	保留。
2	0	只写	停用 LVR 功能。0 / 1：启用 / 停用
1 - 0	00	只写	看门狗时钟超时时间设定： 00：8K 个 ILRC 时钟周期 01：16K 个 ILRC 时钟周期 10：64K 个 ILRC 时钟周期 11：256K 个 ILRC 时钟周期



### 6.3. 看门狗超时溢出复位

看门狗是一个计数器，其时钟源来自 ILRC，所以当 ILRC 关闭时，看门狗也会失效。ILRC 的频率有可能因为工厂制造的变化，电源电压和工作温度而漂移很多，使用者必须预留安全操作范围。

为确保看门狗在超时溢出之前被清零，在安全时间内，可以用指令 `wdreset` 清零看门狗。在上电复位 (POR) 或任何时候使用 `wdreset` 指令，看门狗都会被清零。

当看门狗超时溢出时，JZ8M117 将复位并重新运行程序，其复位时序图如图 5 所示。发生 WDT 复位时，JZ8M117 数据存储器的值将被保留。

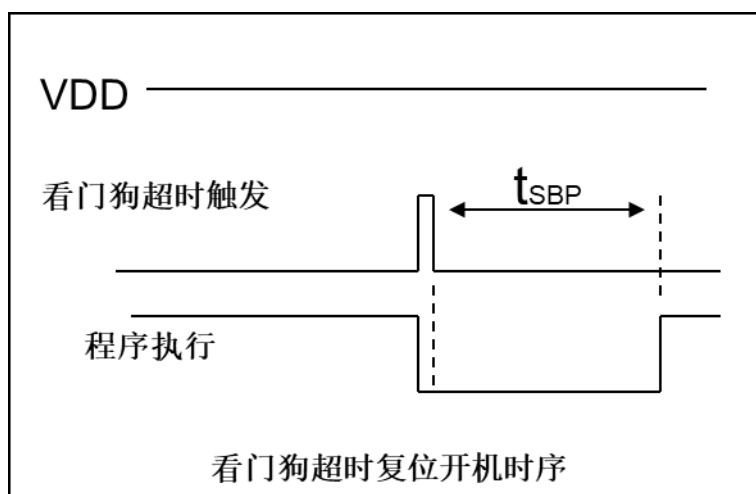


图 5: 看门狗超时溢出的相关时序

利用寄存器 `MISC[1:0]` 可选择四种不同的看门狗超时时间，利用 `CLKMD.1` 可以选择将看门狗功能停用。

时钟控制寄存器(CLKMD)，地址 = 0x03			
位	初始值	读/写	描述
7 - 5	111	读/写	系统时钟选择
4	1	读/写	内部高频 RC 振荡器功能。 0/1: 停用/启用
3	0	读/写	时钟类型选择
2	1	读/写	内部低频 RC 振荡器功能。 0/1: 停用/启用 当 ILRC 关闭时，看门狗也会失效
1	1	读/写	看门狗功能。 0/1: 停用/启用
0	0	读/写	引脚 PA5/PRSTB 功能。 0 / 1: PA5 / PRSTB



### 6.4. 外部复位(PRSTB)

JZ8M117 支持外部复位功能，其外部复位引脚与 PA5 共享同一个 IO 端口。使用外部复位功能需要：

- (1) 设定 PA5 为输入；
- (2) 设定 CLKMD.0=1，使 PA5 为外部 PRSTB 输入脚位。

在外部复位引脚为高电平时，系统处于正常工作状态；一旦复位引脚检测到低电平，系统即发生复位。PRSTB 复位时序图如图 6 所示。

当发生 PRSTB 复位时，JZ8M117 数据存储器的值将被保留。

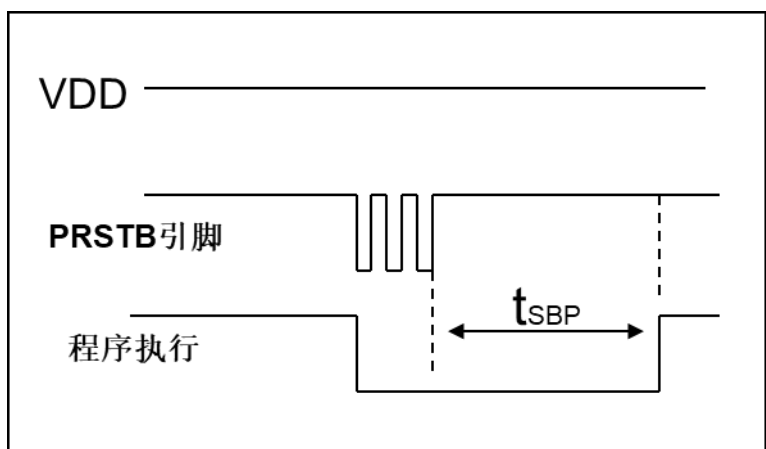


图 6：外部引脚复位的相关时序

### 6.5. 系统电压检测器 - LVD

JZ8M117 内置一个 VDD 电压检测器，用于快速检测芯片的工作电压水平。

通过寄存器 LVDC[7:4]，用户可以从 1.8V 到 4.5V 共 16 个电平中选择一个来与 VDD 电压进行比较。通过读取 LVDC.0 的比较结果，用户就可以监控系统电源是否在预设电平之内。

#### 6.5.1. 低电压检测寄存器(LVDC), 地址 = 0x2D

位	初始值	读/写	描述
7 - 4	0000	WO	LVD 电压水平选择 0000 : 1.8V 0001 : 1.9V 0010 : 2.0V 0011 : 2.1V 0100 : 2.2V 0101 : 2.3V 0110 : 2.4V 0111 : 2.5V 1000 : 2.75V 1001 : 3.0V 1010 : 3.15V 1011 : 3.3V 1100 : 3.5V 1101 : 3.75V 1110 : 4.0V 1111 : 4.5V
3 - 1	-	-	保留
0	-	RO	VDD 电压检测结果 0: VDD > LVD 1: VDD < LVD





## 7. 系统工作模式

JZ8M117 有三个由硬件定义的操作模式，分别为：

- (1) 正常工作模式
- (2) 电源省电模式
- (3) 掉电模式

正常工作模式是所有功能都正常运行的状态；

省电模式(*stopexe*)是在降低工作电流而且 CPU 保持在随时可以继续工作的状态； 掉电模式(*stopsys*)是用来深度的节省电力。

省电模式适合在偶尔需要唤醒的系统工作，掉电模式是在非常低消耗功率且很少需要唤醒的系统中使用。

### 7.1. 省电模式(“*stopexe*”)

使用 *stopexe* 指令进入省电模式，只有系统时钟被停用，其余所有的振荡器模块都继续工作。所以只有 CPU 是停止执行指令。输入引脚切换引起的系统唤醒可视为单片机继续正常的运行。

省电模式的详细信息如下所示：

- (1) IHRC 和振荡器模块：没有变化。如果它被启用，它仍然继续保持活跃。
- (2) ILRC 振荡器模块：必须保持启用，唤醒时需要靠 ILRC 启动。
- (3) 系统时钟停用，因此，CPU 停止执行。
- (4) MTP 存储器被关闭。
- (5) Timer 计数器：若 Timer 计数器的时钟源是系统时钟或其相应的时钟振荡器模块被停用，则 Timer 停止计数；否则，仍然保持计数。（其中，Timer 包含 Timer16, TM2, TM3。）
- (6) 唤醒来源：
  - a. IO Toggle 唤醒：IO 在数字输入模式下的电平变换（*PxC* 位是 0，*PxDIER* 位是 1）。
  - b. Timer 唤醒：如果计数器（Timer）的时钟源不是系统时钟，则当计数到设定值时，系统会被唤醒。
  - c. 比较器唤醒：使用比较器唤醒时，需同时设定 *GPCC.7* 为 1 与 *GPCS.6* 为 1 来启用比较器唤醒功能。但请注意：内部 1.20V Bandgap 参考电压不适用于比较器唤醒功能。

以下例子是利用 Timer16 来唤醒系统因 *stopexe* 的省电模式：

```
$ T16M  ILRC, /1, BIT8           // Timer16 设置
...
WORD    count
        =
        0; STT16
        count;
stopexe;
...           //Timer16 的初始值为 0，在 Timer16 计数了 256 个 ILRC 时钟后，系统将被唤醒。
```



### 7.2. 掉电模式(“*stopsys*”)

掉电模式是深度省电的状态，所有的振荡器模块都会被关闭。使用 *stopsys* 指令可以使芯片直接进入掉电模式。在下达 *stopsys* 指令之前建议将 *GPCC.7* 设为 0 来关闭比较器。

输入引脚的唤醒可视为程序正常运行，为了降低功耗，进入掉电模式之前，所有的 I/O 引脚应仔细检查，避免悬空而漏电。

下面显示发出 *stopsys* 命令后，JZ8M117 内部详细的状态：

- (1) 所有的振荡器模块被关闭。
- (2) MTP 存储器被关闭。
- (3) SRAM 和寄存器内容保持不变。
- (4) 唤醒源：IO 在数字输入模式下电平变换（*PxDIER* 位是 1）。

掉电模式参考示例程序如下所示：

```
CLKMD = 0xF4;           // 系统时钟从 IHRC 变为 ILRC，关闭看门狗时钟
CLKMD.4 = 0;            // IHRC 停用
...
while (1)
{
    stopsys;             // 进入掉电模式
    if (...) break;      // 假如发生唤醒而且检查 OK，就返回正常工作
                        // 否则，停留在掉电模式。
}
CLKMD = 0x34;           // 系统时钟从 ILRC 变为 IHRC/2
```

### 7.3. 唤醒

进入掉电或省电模式后，JZ8M117 可以通过切换 IO 引脚恢复正常工作。而 Timer 的唤醒只适用于省电模式。表 6 显示 *stopsys* 掉电模式和 *stopexe* 省电模式在唤醒源的差异。

掉电模式( <i>stopsys</i> )和省电模式 ( <i>stopexe</i> )在唤醒源的差异			
	切换 IO 引脚	计时器唤醒	比较器唤醒
<i>stopsys</i>	是	否	否
<i>stopexe</i>	是	是	是

表 6：掉电模式和省电模式在唤醒源的差异



当使用 IO 引脚来唤醒 JZ8M117，寄存器 *PxDIER* 应正确设置，使每一个相应的引脚可以有唤醒功能。从唤醒事件发生后开始计数，正常的唤醒时间大约是 3000 个 ILRC 时钟周期，另外，JZ8M117 提供快速唤醒功能，透过

*MISC* 寄存器选择快速唤醒大约 45 个 ILRC 时钟周期。

模式	唤醒模式	切换 IO 引脚的唤醒时间( <i>twup</i> )
STOPEXE 省电模式 STOPSYS 掉电模式	快速唤醒	$45 * T_{ILRC}$ , 这里的 $T_{ILRC}$ 是指 ILRC 时钟周期
STOPEXE 省电模式 STOPSYS 掉电模式	正常唤醒	$3000 * T_{ILRC}$ , 这里的 $T_{ILRC}$ 是指 ILRC 时钟周期

表 7：快速唤醒和正常唤醒的时间差异

## 8. 中断

JZ8M117 有 8 个中断源：

- ◆ 外部中断源 PA0/PA5
- ◆ 外部中断源 PB0
- ◆ Timer16 中断源
- ◆ Timer2 中断源
- ◆ Timer3 中断源
- ◆ 比较器中断源
- ◆ 2 个触摸按键中断(TK\_OV 和 TK\_END)。

中断硬件框图请参考图 7。每个中断请求源都有自己的中断控制位启用或停用它。所有的中断请求标志位是由硬件置位并且并通过软件写寄存器 *INTRQ* 清零。中断请求标志设置点可以是上升沿或下降沿或两者兼而有之，这取决于对寄存器 *INTEGS* 的设置。所有的中断请求源最后都需由 *engint* 指令控制（启用全局中断）使中断运行，以及使用 *disgint* 指令（停用全局中断）停用它。

中断堆栈是共享数据存储器，其地址由堆栈寄存器 *SP* 指定。由于程序计数器是 16 位宽度，堆栈寄存器 *SP* 位 0 应保持 0。此外，用户可以使用 *pushaf* 指令存储 *ACC* 和标志寄存器的值到堆栈，以及使用 *popaf* 指令将值从堆栈恢复到 *ACC* 和标志寄存器中。由于堆栈与数据存储器共享，在 Mini-C 模式，堆栈位置与深度由编译程序安排。在汇编模式或自行定义堆栈深度时，用户应仔细安排位置，以防地址冲突。

在中断服务程序中，可以通过读取寄存器 *INTRQ* 知道中断发生源。

注：可在 Code Option Interrupt Src0 或 Interrupt Src1 中切换外部中断源。

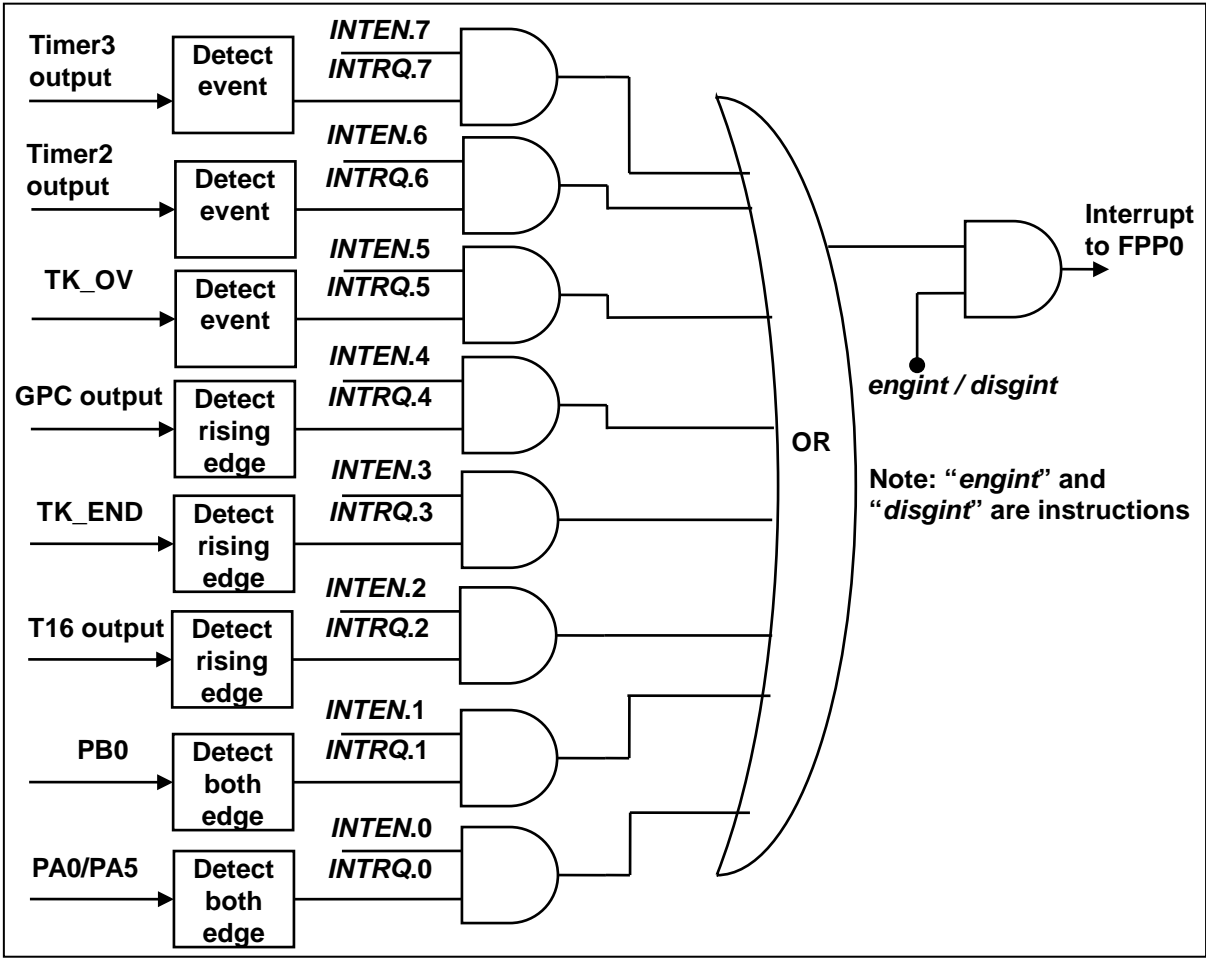


图 7：中断硬件框图

8.1. 中断允许寄存器(INTEN), 地址 = 0x04

位	初始值	读/写	描述
7	0	读/写	启用从 Timer3 的溢出中断。0/1：停用/启用
6	0	读/写	启用从 Timer2 的溢出中断。0/1：停用/启用
5	0	读/写	启用从触摸按键的 TK_OV 中断。0/1：停用/启用
4	0	读/写	启用从比较器的溢出中断。0/1：停用/启用
3	0	读/写	启用从触摸按键的 TK_END 中断。0/1：停用/启用
2	0	读/写	启用从 Timer16 的溢出中断。0/1：停用/启用
1	0	读/写	启用从 PB0 的中断。0/1：停用/启用
0	0	读/写	启用从 PA0/PA5 的中断。0/1：停用/启用



### 8.2. 中断请求寄存器(INTRQ), 地址 = 0x05

位	初始值	读/写	描述
7	-	读/写	Timer3 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
6	-	读/写	Timer2 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
5	-	读/写	触摸按键 TK_OV 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
4	-	读/写	比较器的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
3	-	读/写	触摸按键 TK_END 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
2	-	读/写	Timer16 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
1	-	读/写	PB0 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求
0	-	读/写	PA0/PA5 的中断请求, 此位是由硬件置位并由软件清零。 0/1: 不要求/请求

注意: *INTEN*, *INTRQ* 没有初始值, 所以要使用中断前, 一定要根据需要设定数值。即使 *INTEN* 为 0, *INTRQ* 还是会被中断发生源触发。

### 8.3. 中断边缘选择寄存器(INTEGS), 地址 = 0x0C

Bit	Reset	R/W	Description
7 - 5	-	-	保留, 请设为 0。
4	-	WO	Timer16 中断边缘选择: 0: 上升缘请求中断。 1: 下降缘请求中断。
3 - 2	-	WO	PB0 中断边缘选择: 00: 上升缘和下降缘都请求中断。 01: 上升缘请求中断。 10: 下降缘请求中断。 11: 保留。
1 - 0	-	WO	PA0 / PA5 中断边缘选择: 00: 上升缘和下降缘都请求中断。 01: 上升缘请求中断。 10: 下降缘请求中断。 11: 保留。



### 8.4. 中断工作流程

一旦发生中断，其具体工作流程如下：

- (1) 程序计数器将自动存储到 *SP* 寄存器指定的堆栈存储器。
- (2) 新的 *SP* 将被更新为 *SP*+2。
- (3) 全局中断将自动被停用。
- (4) 将从地址 *0x010* 获取下一条指令。

中断完成后，发出 *reti* 指令返回既有的程序，其具体工作流程如下：

- (1) 从 *SP* 寄存器指定的堆栈存储器自动恢复程序计数器。
- (2) 新的 *SP* 将被更新为 *SP*-2。
- (3) 全局中断将自动启用。
- (4) 下一条指令将是中断前原来的指令。

### 8.5. 中断的一般步骤

步骤 1：设定 *INTEN* 寄存器，开启需要的中断的控制位。

步骤 2：清除 *INTRQ* 寄存器。

步骤 3：主程序中，使用 *engint* 指令（启用全局中断）允许 CPU 的中断功能。

步骤 4：等待中断。中断发生后，跳入中断子程序。

步骤 5：当中断子程序执行完毕，返回主程序。

跳入中断子程序处理时，可使用 *pushaf* 指令来保存 *ALU* 和 *FLAG* 寄存器数据，并在 *reti* 之前，使用 *popaf* 指令复原。一般步骤如下：

```
void Interrupt(void)    // 中断发生后，跳入中断子程序，
{                        // 自动进入 disgint 的状态，CPU 不会再接受中断

    PUSHAF;
    ...
    P
    O
    P
    A
    F;

}                        // 系统自动填入 reti，直到执行 reti 完毕才自动恢复到 engint 的状态
```

\* 在主程序中，可使用 *disgint* 指令关闭所有中断。



## 8.6. 使用中断举例

使用者必须预留足够的堆栈存储器以保存中断向量，一级中断需要两个字节，两级中断需要四个字节。下面的示例程序演示了如何处理中断，请注意，处理中断和 *pushaf* 是需要四个字节堆栈存储器。

```
void    FPPA0    (void)
{
    ...
    $ INTEN PA0;      // INTEN =1; 当 PA0 准位改变，产生中断请求
    INTRQ = 0;        // 清除 INTRQ
    ENGINT            // 启用全局中断

    ...
    DISGINT          // 停用全局中断

    ...
}

void Interrupt (void)    // 中断程序
{
    PUSHAF              // 存储 ALU 和 FLAG 寄存器

    // 如果 INTEN.PA0 在主程序会动态开和关，则表达式中可以判断 INTEN.PA0 是否为 1。
    // 例如： If (INTEN.PA0 && INTRQ.PA0) {...}

    // 如果 INTEN.PA0 一直在使能状态，就可以省略判断 INTEN.PA0，以加速中断执行。

    If (INTRQ.PA0)
    {
        // PA0 的中断程序
        INTRQ.PA0 = 0; // 只须清除相对应的位 (PA0)

        ...
    }

    ...
    // (X:) INTRQ = 0;    // 不建议在中断程序最后，才使用 INTRQ = 0 一次全部清除
                        // 因为它可能会把刚发生而尚未处理的中断，意外清除掉

    POPAF              // 回复 ALU 和 FLAG 寄存器
}
```



## 9. I/O 端口

### 9.1. IO 相关寄存器

#### 9.1.1. 端口 A 数字输入启用寄存器(PADIER), 地址 = 0x0D

位	初始值	读/写	描述
7 - 6	11	只写	使能 PA7~PA6 数字输入和唤醒事件。 1 / 0: 启用/ 停用 当使用外部晶体振荡器的时候, 该位设为 0 防止耗电。
5	1	只写	使能 PA5 数字输入、唤醒事件和中断请求。 1 / 0: 启用/ 停用
4 - 3	11	只写	使能 PA4 ~ PA3 数字输入和唤醒事件。 1 / 0: 启用/ 停用
2 - 1	-	-	保留。
0	1	只写	使能 PA0 数字输入、唤醒事件和中断请求。 1 / 0: 启用 / 停用

#### 9.1.2. 端口 B 数字输入启用寄存器(PBDIER), 地址 = 0x0E

位	初始值	读/写	描述
7	1	只写	使能 PB7 数字输入和唤醒事件。 0 / 1: 停用 / 启用
6 - 1	-	-	保留
0	1	只写	使能 PB0 数字输入、唤醒事件和中断请求。 0 / 1: 停用 / 启用

#### 9.1.3. 端口 A 数据寄存器(PA), 地址 = 0x10

位	初始值	读/写	描述
7 - 0	0x00	读/写	数据寄存器的端口 A。

#### 9.1.4. 端口 A 控制寄存器(PAC), 地址 = 0x11

位	初始值	读/写	描述
7 - 0	0x00	读/写	端口 A 控制寄存器。这些寄存器是用来定义端口 A 每个相应引脚的输入模式或输出模式。 0 / 1: 输入/输出

#### 9.1.5. 端口 A 上拉控制寄存器(PAPH), 地址 = 0x12

位	初始值	读/写	描述
7 - 0	0x00	读/写	端口 A 上拉控制寄存器。该寄存器是用来控制端口 A 每个相应引脚的上拉功能。 0/1: 停用/启用





### 9.1.6. 端口 A 下拉控制寄存器(PAPL), 地址 = 0x13

位	初始值	读/写	描 述
7 - 0	0x00	读/写	端口 A 下拉控制寄存器。该寄存器是用来控制端口 A 每个相应引脚的下拉功能。 0/1: 停用/启用

### 9.1.7. 端口 B 数据寄存器(PB), 地址 = 0x14

位	初始值	读/写	描 述
7 - 0	0x00	读/写	数据寄存器的端口 B。

### 9.1.8. 端口 B 控制寄存器(PBC), 地址 = 0x15

位	初始值	读/写	描 述
7 - 0	0x00	读/写	端口 B 控制寄存器。这些寄存器是用来定义端口 B 每个相应的引脚的输入模式或输出模式。0/1: 输入/输出

### 9.1.9. 端口 B 上拉控制寄存器(PBPH), 地址 = 0x16

位	初始值	读/写	描 述
7 - 0	0x00	读/写	端口 B 上拉控制寄存器。这些寄存器是用来控制端口 B 每个相应引脚上拉电阻的使能。 0/1: 停用/启用

### 9.1.10. 端口 B 下拉控制寄存器(PBPL), 地址 = 0x17

位	初始值	读/写	描 述
7 - 0	0x00	读/写	端口 B 下拉控制寄存器。这些寄存器是用来控制端口 B 每个相应引脚下拉电阻的使能。 0/1: 停用/启用



## 9.2. IO 结构及功能

### 921. IO 引脚的结构

JZ8M117 的所有 IO 引脚都具有相同的结构，如下图 8。

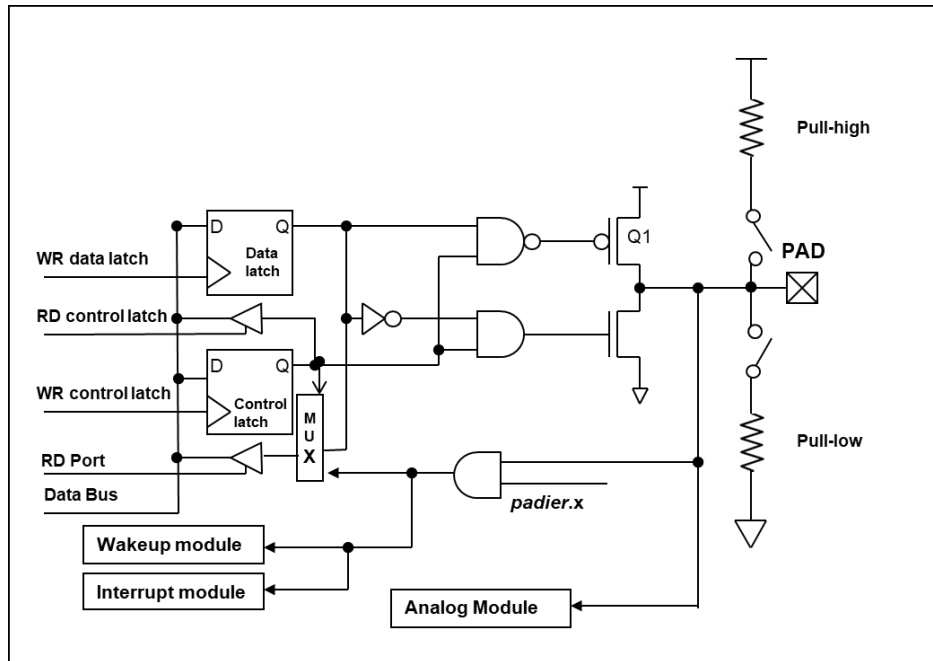


图 8: 引脚缓冲区硬件图

### 922 IO 引脚的一般功能

#### (1)输入、输出功能:

JZ8M117 所有 IO 引脚都可编程设定为数字输入或模拟输入、低输出或高输出。

透过数据寄存器(PA/PB)，控制寄存器(PAC/PBC)和上拉、下拉控制寄存器(PAPH/PBPH、PAPL/PBPL) 设定，每一 IO 引脚都可以独立配置成不同的功能。

当引脚被用做模拟输入功能时，为减少漏电流，请关闭 *PxDIER* 寄存器相应位的数字输入功能。当这些引脚为输出状态时，其弱上拉/下拉电阻会自动关闭。

如果要读取端口上的电位状态，一定要先将端口设置成输入模式；在输出模式下，读取到的数据是数据寄存器的值。表 8 为端口 PA0 的设定配置表。

PA.0	PAC.0	PAPH.0	PAPL.0	描述
X	0	0	0	输入悬空，没有上拉/下拉电阻
X	0	1	0	输入，有上拉电阻
X	0	0	1	输入，有下拉电阻
X	0	1	1	输入，有上拉/下拉电阻
0	1	X	X	输出低电位
1	1	X	X	输出高电位

表 8: PA0 设定配置表



### (2)睡眠唤醒功能:

当 JZ8M117 在掉电或省电模式，每一个引脚都可以切换其状态来唤醒系统。对于需用来唤醒系统的引脚，必须设置为输入模式以及寄存器 *PxDIER* 相应位为高。

### (3)外部中断功能:

当 IO 作为外部中断引脚时，*PxDIER* 相应位应设置高。例，当 PA0 用来作为外部中断引脚时，*PADIER.0* 应设置高。

### (4)驱动能力可选:

部分引脚可通过程序选项 *Drive* 来调整驱动电流和灌电流。

## 923. IO 使用与设定

### (1) IO 作为数字输入

- ◆ 将 IO 设为数字输入时，*Vih* 与 *Vil* 的准位，会随着工作电压和温度有变动。请参考 *Vih* 最小值和 *Vil* 最大值。
- ◆ 内部上拉电阻值也将随着电压、温度与引脚电压而变动，并非为固定值。

### (2) IO 作为数字输入和打开唤醒功能

- ◆ 用 *PxC* 寄存器，将 IO 设为输入。
- ◆ 用 *PxDIER* 寄存器，将对应的位设为 1 以启用数字输入。
- ◆ 为了防止 PA 中没有用到的 IO 口漏电，*PADIER*[1:2]需要常设为 0。

### (3) PA5 作为 PRSTB 输入

- ◆ 设定 PA5 为输入。
- ◆ 设定 *CLKMD.0*=1，使 PA5 为外部 PRSTB 输入脚位。

### (4) PA5 作为输入并通过长导线连接至按键或者开关

- ◆ 必需在 PA5 与长导线中间串接  $>10\ \Omega$  电阻。
- ◆ 应尽量避免使用 PA5 作为输入。



## 10. Timer / PWM 计数器

### 10.1. 16 位计数器 (Timer16)

#### 10.1.1. Timer16 介绍

JZ8M117 内置一个 16 位硬件计数器 Timer16(T16)，其模块框图如图 9。

计数器时钟源由寄存器 *T16M*[7:5]来选择，在时钟送到 16 位计数器(counter16)之前，*T16M*[4:3]可对时钟进行预分频处理，有÷1、÷4、÷16、÷64 等四种选项，让计数范围更大。

*T16M*[2:0]用于选择 Timer16 的中断源，其来自于 16 位计数器的位 8 到位 15。当计数器溢出时，Timer16 就触发中断。经由寄存器 *INTEGS.4*，可选择中断类型是上升沿触发或下降沿触发。

16 位计数器只能向上计数，计数器初始值可以用 *stt16* 指令设定，计数器的数值可以用 *ldt16* 指令存储到数据存储器。

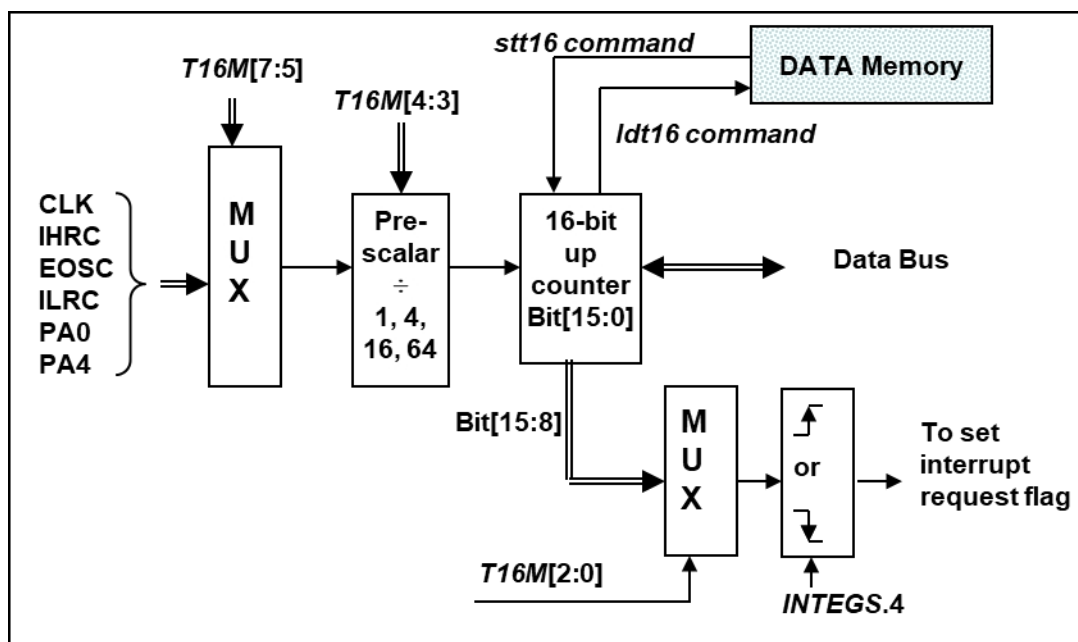


图 9: Timer16 模块框图

Timer16 的语法定义在.inc 文件中。*T16M* 共有三个配置参数，第一个参数用来定义 Timer16 的时钟源，第二个参数用来定义预分频器，第三个参数是确定中断源。

```
T16M IO_RW 0x06
$ 7~5: STOP, SYSCLK, X, PA4_F, IHRC, EOSC, ILRC, PA0_F // 第一个参数
$ 4~3: /1, /4, /16, /64 // 第二个参数
$ 2~0: BIT8, BIT9, BIT10, BIT11, BIT12, BIT13, BIT14, BIT15 // 第三个参数
```



使用者可以依照系统的要求来定义 *T16M* 参数，例子如下：

**\$ T16M      SYSCLK, /64, BIT15;**

// 选择(SYSCLK/64) 当 Timer16 时钟源，每  $2^{16}$  个时钟周期产生一次 INTRQ.2=1

// 系统时钟 System Clock = IHRC / 2 = 8 MHz

// SYSCLK/64 = 8 MHz/64 = 8 uS，约每 524 mS 产生一次 INTRQ.2=1

**\$ T16M      PA0, /1, BIT8;**

// 选择 PA0 当 Timer16 时钟源，每  $2^9$  个时钟周期产生一次 INTRQ.2=1

// 每接收 512 个 PA0 时钟周期产生一次 INTRQ.2=1

**\$ T16M      STOP;**

// 停止 Timer16 计数

#### 10.1.2. Timer16 控制寄存器(T16M)，地址 = 0x06

位	初始值	读/写	描 述
7 – 5	000	读/写	Timer16 时钟选择： 000: 停用 Timer16 001: CLK 系统时钟 010: 保留 011: PA4 下降沿（外部事件） 100: IHRC 101: EOSC 110: ILRC 111: PA0 下降沿（外部事件）
4 – 3	00	读/写	Timer16 内部的时钟分频器。 00: ÷1 01: ÷4 10: ÷16 11: ÷64
2 – 0	000	读/写	中断源选择。当选择位由低变高或由高变低时，发生中断事件。 0: Timer16 位 8 1: Timer16 位 9 2: Timer16 位 10 3: Timer16 位 11 4: Timer16 位 12 5: Timer16 位 13 6: Timer16 位 14 7: Timer16 位 15



### 10.1.3. Timer16 溢出时间

当设定  $\$INTEGS\ BIT\_R$  时（这是 IC 默认值），且设定  $T16M$  计数器 BIT8 产生中断，若 T16 计数从 0 开始，则第一次中断是在计数到 0x100 时发生（BIT8 从 0 到 1），第二次中断在计数到 0x300 时发生（BIT8 从 0 到 1）。所以设定 BIT8 是计数 512 次才中断。请注意，如果在中断中重新给  $T16M$  计数器设值，则下一次中断也将在 BIT8 从 0 变 1 时发生。

如果设定  $\$INTEGS\ BIT\_F$ （BIT 从 1 到 0 触发）而且设定  $T16M$  计数器 BIT8 产生中断，则 T16 计数改为每次数到 0x200/0x400/0x600/...时发生中断。两种设定  $INTEGS$  的方法各有好处，也请注意其中差异。

### 10.2. 8 位 PWM 计数器(Timer2,Timer3)

JZ8M117 内置 2 个 8 位 PWM 硬件定时器(Timer2/TM2,Timer3/TM3)，两个计数器的原理一样，以下以 Timer2 来说明，TM2 硬件框图请参考图 10。

寄存器  $TM2C[7:4]$ 用来选择定时器时钟； $TM2C[3:2]$ 用来选择 Timer2 的输出。寄存器  $TM2S[6:0]$ 用于选择时钟分频处理。寄存器  $TM2B$  用来控制定时器的计数上限，当计数值达到  $TM2B$  设定的上限时，定时器将自动清零。寄存器  $TM2CT$  用于设置或读取定时器的计数值。

8 位 PWM 定时器的工作模式有周期模式和 PWM 模式两种。周期模式用于输出固定周期波形；PWM 模式是用来产生 PWM 输出波形，PWM 分辨率可以为 6~8 位。

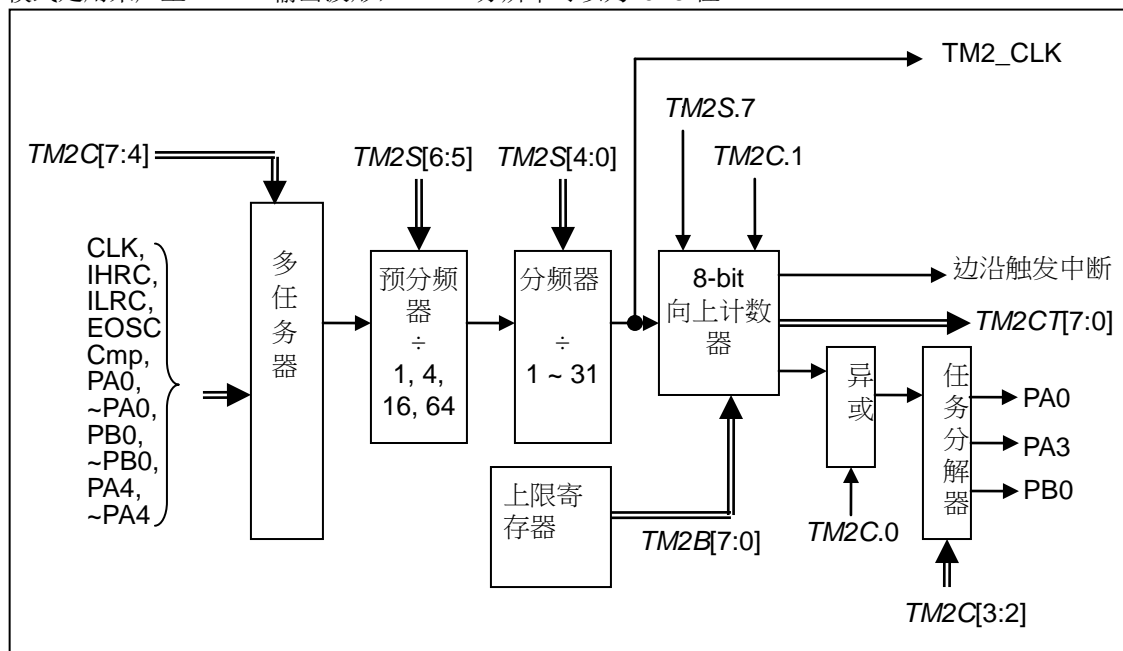


图 10: Timer2 模块框图

Timer3 的计数输出可选择为 PA4, PA5 或 PB7。



图 11 显示出 Timer2 周期模式和 PWM 模式的时序图：

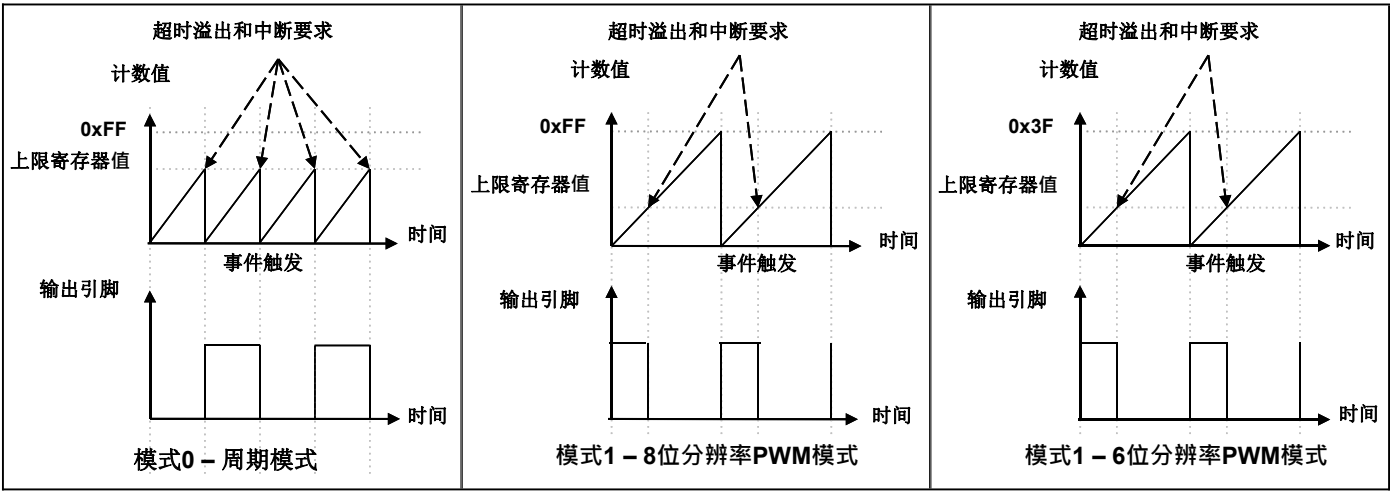


图 11: Timer2 周期模式和 PWM 模式的时序图

程序选项”GPC\_PWM“是指由比较器结果控制 PWM 波形。选用此功能后，当比较器输出为 1 时，PWM 停止输出；比较器输出为 0 时，PWM 恢复输出。如图 12 所示。

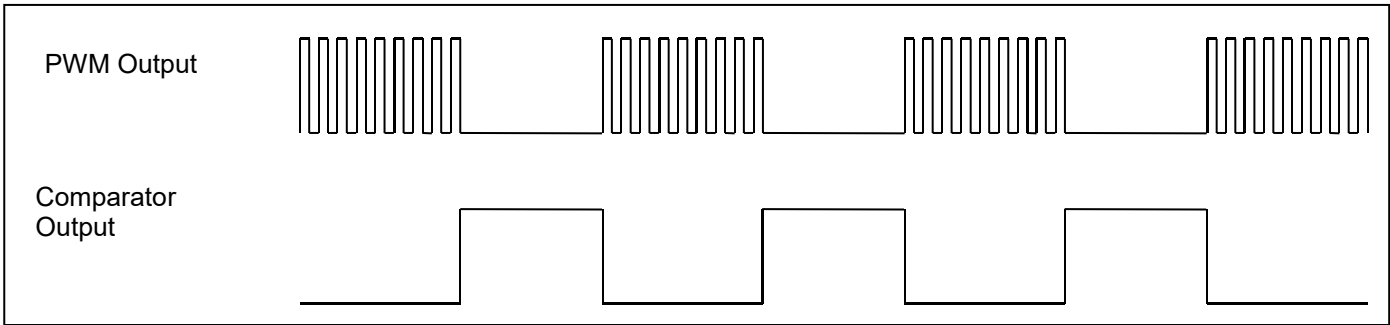


图 12: 比较器控制 PWM 输出

10.2.1. Timer2、Timer3 相关寄存器

10.2.1.1. Timer2 分频寄存器(TM2S)，地址 = 0x1E

位	初始值	读/写	描述
7	0	只写	PWM 分辨率选择。 0: 8 位 1: 6 位或者 7 位（由程序选项 TMx_bit 控制）
6 – 5	00	只写	Timer2 时钟预分频器。 00: ÷ 1 01: ÷ 4 10: ÷ 16 11: ÷ 64
4 – 0	00000	只写	Timer2 时钟分频器。



### 10.2.1.2. Timer2 控制寄存器(TM2C), 地址 = 0x1C

位	初始值	读/写	描 述
7 – 4	0000	读/写	Timer2 时钟源选择。 0000: 停用 0001: CLK 0010: IHRC 或者 IHRC*2 (由程序选项 TMx_source 控制) 0011: EOSC 0100: ILRC 0101: 比较器输出 1000: PA0 (上升沿) 1001: ~PA0 (下降沿) 1010: PB0 (上升沿) 1011: ~PB0 (下降沿) 1100: PA4 (上升沿) 1101: ~PA4 (下降沿) 其他: 保留 <u>注意:</u> 在 ICE 模式且 IHRC 被选为 Timer2 定时器时钟, 当 ICE 停下时, 发送到定时器的时钟是不停止, 定时器仍然继续计数。
3 – 2	00	读/写	Timer2 输出选择。 00: 停用 01: PA0 10: PA3 11: PB0
1	0	读/写	Timer2 模式选择。 0 / 1: 周期模式 / PWM 模式。
0	0	读/写	启用 Timer2 反极性输出。 0 / 1: 停用/启用

### 10.2.1.3. Timer2 计数寄存器(TM2CT), 地址 = 0x1D

位	初始值	读/写	描 述
7 – 0	0x00	读/写	Timer2 定时器位[7:0]。

### 10.2.1.4. Timer2 上限寄存器(TM2B), 地址 = 0x1F

位	初始值	读/写	描 述
7 – 0	0x00	只写	Timer2 上限寄存器。

### 10.2.1.5. Timer3 计数寄存器(TM3CT), 地址 = 0x33

位	初始值	读/写	描 述
7 – 0	0x00	读/写	Timer3 定时器位[7:0]。





# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

### 10.2.1.6. Timer3 分频寄存器(TM3S)，地址= 0x34

位	初始值	读/写	描 述
7	0	只写	PWM 分辨率选择。 0: 8 位 1: 6 位或 7 位（由程序选项 TMx_bit 控制）
6 – 5	00	只写	Timer3 时钟预分频器。 00: ÷ 1 01: ÷ 4 10: ÷ 16 11: ÷ 64
4 – 0	00000	只写	Timer3 时钟分频器。

### 10.2.1.7. Timer3 上限寄存器(TM3B)，地址 = 0x35

位	初始值	读/写	描 述
7 – 0	0x00	只写	Timer3 上限寄存器。

### 10.2.1.8. Timer3 控制寄存器(TM3C)，地址 = 0x32

位	初始值	读/写	描 述
7 – 4	0000	读/写	Timer3 时钟选择。 0000: 停用 0001: CLK 0010: IHRC 或者 IHRC*2（由程序选项 TMx_source 控制） 0011: EOSC 0100: ILRC 0101: 比较器输出 1000: PA0（上升沿） 1001: ~PA0（下降沿） 1010: PB0（上升沿） 1011: ~PB0（下降沿） 1100: PA4（上升沿） 1101: ~PA4（下降沿） 其他: 保留 <u>注意:</u> 在 ICE 模式且 IHRC 被选为 Timer3 定时器时钟，当 ICE 停下时，发送到定时器的时钟是不停止，定时器仍然继续计数。
3 – 2	00	读/写	Timer3 输出选择。 00: 停用 01: PA4 10: PA5 11: PB7
1	0	读/写	Timer3 模式选择。 0: 周期模式 1: PWM 模式
0	0	读/写	启用 Timer3 反极性输出。 0 / 1: 停用/启用。



### 10.2.2 使用 Timer2 产生定期波形

如果选择周期模式的输出，输出波形的占空比总是 50%，其输出频率与寄存器设定，可以概括如下：

$$\text{输出频率} = Y \div [2 \times (K+1) \times S1 \times (S2+1)]$$

这里，

$Y = TM2C[7:4]$  : Timer2 所选择的时钟源频率

$K = TM2B[7:0]$  : 上限寄存器设定的值（十进制）

$S1 = TM2S[6:5]$  : 预分频器设定值 ( $S1 = 1, 4, 16, 64$ )

$S2 = TM2S[4:0]$  : 分频器值（十进制， $S2 = 0 \sim 31$ ）

例 1:

$TM2C = 0b0001\_1100$ ,  $Y=8MHz$

$TM2B = 0b0111\_1111$ ,  $K=127$

$TM2S = 0b0\_00\_00000$ ,  $S1=1$ ,  $S2=0$

→ 输出频率 =  $8MHz \div [2 \times (127+1) \times 1 \times (0+1)] = 31.25KHz$

例 2:

$TM2C = 0b0001\_1100$ ,  $Y=8MHz$

$TM2B = 0b0000\_0001$ ,  $K=1$

$TM2S = 0b0\_00\_00000$ ,  $S1=1$ ,  $S2=0$

→ 输出频率 =  $8MHz \div [2 \times (1+1) \times 1 \times (0+1)] = 2MHz$

使用 Timer2 定时器产生定期波形的示例程序如下所示：

```
void FPPA0 (void)
{
    . ADJUST_IC  SYSCLK=IHRC/2, IHRC=16MHz, VDD=5V
    ...
    TM2CT = 0x00;
    TM2B = 0x7f;
    TM2S = 0b0_00_00001;           // 8 位 PWM, 预分频 = 1, 分频 = 2
    TM2C = 0b0001_10_0_0;         // 系统时钟, 输出 =PA3, 周期模式
    while(1)
    {
        nop;
    }
}
```



### 10.2.3 使用 Timer2 产生 8 位 PWM 波形

如果选择 8 位 PWM 的模式，应设立  $TM2C.1 = 1$ ,  $TM2S.7 = 0$ ，输出波形的频率和占空比可概括如下：

$$\text{输出频率} = Y \div [256 \times S1 \times (S2+1)]$$

$$\text{输出空占比} = [(K+1) \div 256] \times 100\%$$

这里，

$Y = TM2C[7:4]$  : Timer2 所选择的时钟源频率

$K = TM2B[7:0]$  : 上限寄存器设定的值（十进制）

$S1 = TM2S[6:5]$  : 预分频器设定值 ( $S1 = 1, 4, 16, 64$ )

$S2 = TM2S[4:0]$  : 分频器值（十进制， $S2 = 0 \sim 31$ ）

例 1:

$TM2C = 0b0001\_1110$ ,  $Y=8MHz$

$TM2B = 0b0111\_1111$ ,  $K=127$

$TM2S = 0b0\_00\_00000$ ,  $S1=1$ ,  $S2=0$

→ 输出频率 =  $8MHz \div (256 \times 1 \times (0+1)) = 31.25KHz$

→ 输出空占比 =  $[(127+1) \div 256] \times 100\% = 50\%$

例 2:

$TM2C = 0b0001\_1110$ ,  $Y=8MHz$

$TM2B = 0b0000\_1001$ ,  $K = 9$

$TM2S = 0b0\_00\_00000$ ,  $S1=1$ ,  $S2=0$

→ 输出频率 =  $8MHz \div (256 \times 1 \times (0+1)) = 31.25KHz$

→ 输出空占比 =  $[(9+1) \div 256] \times 100\% = 3.9\%$

使用 Timer2 定时器产生 PWM 波形的示例程序如下所示：

```
void FPPA0(void)
{
    . ADJUST_IC  SYSCLK=IHRC/2, IHRC=16MHz, VDD=5V
    ...
    TM2CT = 0x00;
    TM2B = 0x7f;
    TM2S = 0b0_00_00001;    //8 位 PWM, 预分频 = 1, 分频 = 2
    TM2C = 0b0001_10_1_0;   //系统时钟, 输出 = PA3, PWM 模式
    while(1)
    {
        nop;
    }
}
```



#### 10.2.4 使用 Timer2 产生 6 位 PWM 波形

如果选择 6 位 PWM 的模式，应设立  $TM2C.1 = 1$ ， $TM2S.7 = 1$ ，输出波形的频率和占空比可概括如下：

$$\text{输出频率} = Y \div [64 \times S1 \times (S2+1)]$$

$$\text{输出空占比} = [(K+1) \div 64] \times 100\%$$

这里，

$Y = TM2C[7:4]$  : Timer2 所选择的时钟源频率

$K = TM2B[7:0]$  : 上限寄存器设定的值（十进制）

$S1 = TM2S[6:5]$  : 预分频器设定值 ( $S1 = 1, 4, 16, 64$ )

$S2 = TM2S[4:0]$  : 分频器值（十进制， $S2 = 0 \sim 31$ ）

例 1:

$TM2C = 0b0001\_1110$ ,  $Y=8MHz$

$TM2B = 0b0011\_1111$ ,  $K=63$

$TM2S = 0b1\_00\_00000$ ,  $S1=1$ ,  $S2=0$

→ 输出频率 =  $8MHz \div (64 \times 1 \times (0+1)) = 125KHz$

→ 输出空占比 =  $[(63+1) \div 64] \times 100\% = 100\%$

例 2:

$TM2C = 0b0001\_1110$ ,  $Y=8MHz$

$TM2B = 0b0000\_0000$ ,  $K=0$

$TM2S = 0b1\_00\_00000$ ,  $S1=1$ ,  $S2=0$

→ 输出频率 =  $8MHz \div (64 \times 1 \times (0+1)) = 125KHz$

→ 输出空占比 =  $[(0+1) \div 64] \times 100\% = 1.5\%$



## 11. 特殊功能

### 11.1. 比较器

JZ8M117 内置一个硬件比较器，硬件框图如图 13。它可以比较两个输入端之间的信号大小。进行比较的两个信号，一个是正输入，另一个是负输入。正输入由寄存器 *GPCC.0* 选择；负输入由 *GPCC[3:1]* 选择。

比较器输出的结果可以：

- (1) 由 *GPCC.6* 读取出来；
- (2) 由 *GPCC.4* 选择输出信号是否反极性；
- (3) 由 *GPCC.5* 选择是否由 Time2(TM2\_CLK)采样输出；
- (4) 由 *GPCS.7* 选择是否输出到 PA0；
- (5) 产生中断信号。

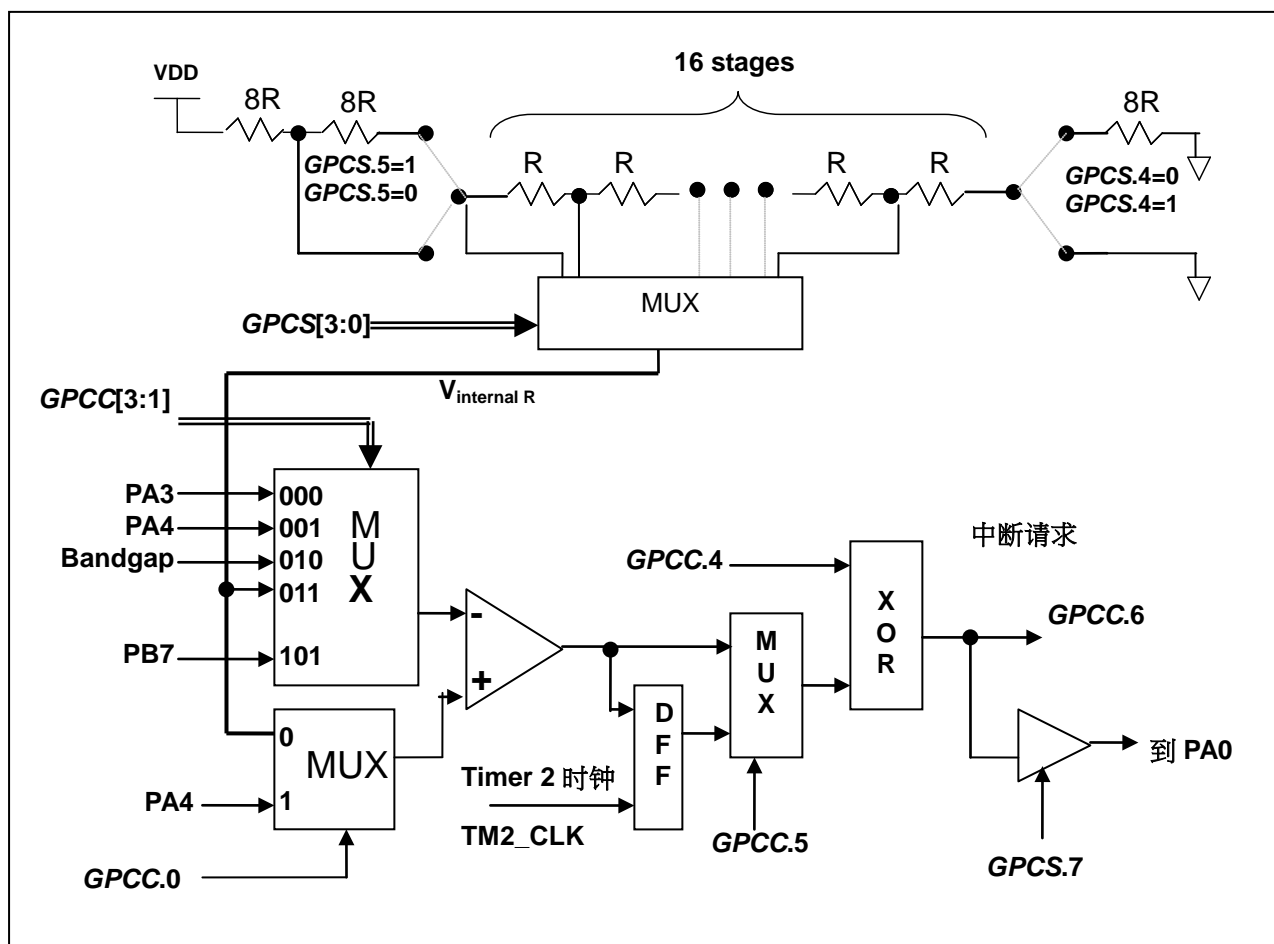


图 13：比较器硬件图框



### 11.1.1. 比较器控制寄存器(GPCC), 地址= 0x18

位	初始值	读/写	描 述
7	0	读/写	启用比较器。0 / 1：停用/启用 当此位被设置为启用，请同时设置相应的模拟输入引脚是数字停用，以防止漏电。
6	-	只读	比较器结果。 0: 正输入 < 负输入 1: 正输入 > 负输入
5	0	读/写	选择比较器的结果是否由 TM2_CLK 采样输出。 0: 比较器的结果没有 TM2_CLK 采样输出 1: 比较器的结果是由 TM2_CLK 采样输出
4	0	读/写	选择比较器输出的结果是否反极性。 0: 比较器输出的结果没有反极性 1: 比较器输出的结果是反极性
3 – 1	000	读/写	选择比较器负输入的来源。 000: PA3 001: PA4 010: 内部 1.20 V Bandgap 参考电压（不适用于比较器唤醒功能） 011: $V_{internal R}$ 100: 保留 101: PB7（仿真器不支持） 11X: 保留
0	0	读/写	选择比较器正输入的来源。 0: $V_{internal R}$ 1: PA4

### 11.1.2. 比较器选择寄存器(GPCS), 地址 = 0x19

位	初始值	读/写	描 述
7	0	只写	比较器输出启用（到 PA0）。 0 / 1: 停用/启用。 （仿真时，如果比较器输出到 PA0，则 PA3 会被设为输出。实际 IC 没有此问题）
6	0	只写	比较器唤醒启用。（gpcc.6 发生电平变化时才可唤醒） 0/1: 停用/启用
5	0	只写	选择比较器参考电压 $V_{internal R}$ 最高的范围。
4	0	只写	选择比较器参考电压 $V_{internal R}$ 最低的范围。
3 – 0	0000	只写	选择比较器参考电压 $V_{internal R}$ 。 0000（最低）~ 1111（最高）

**11.1.3. 内部参考电压 ( $V_{\text{internal R}}$ )**

内部参考电压  $V_{\text{internal R}}$  由一连串电阻组成，可以通过寄存器  $\text{GPCS}[5:0]$  来设置具体数值，范围从  $(1/32)*V_{\text{DD}}$  到  $(3/4)*V_{\text{DD}}$ 。寄存器  $\text{GPCS}$  的位 4 和位 5 用来选择  $V_{\text{internal R}}$  的最高和最低值；位[3:0]用于选择所要的电压水平，这电压水平是由  $V_{\text{internal R}}$  的最高和最低值均分 16 等份。

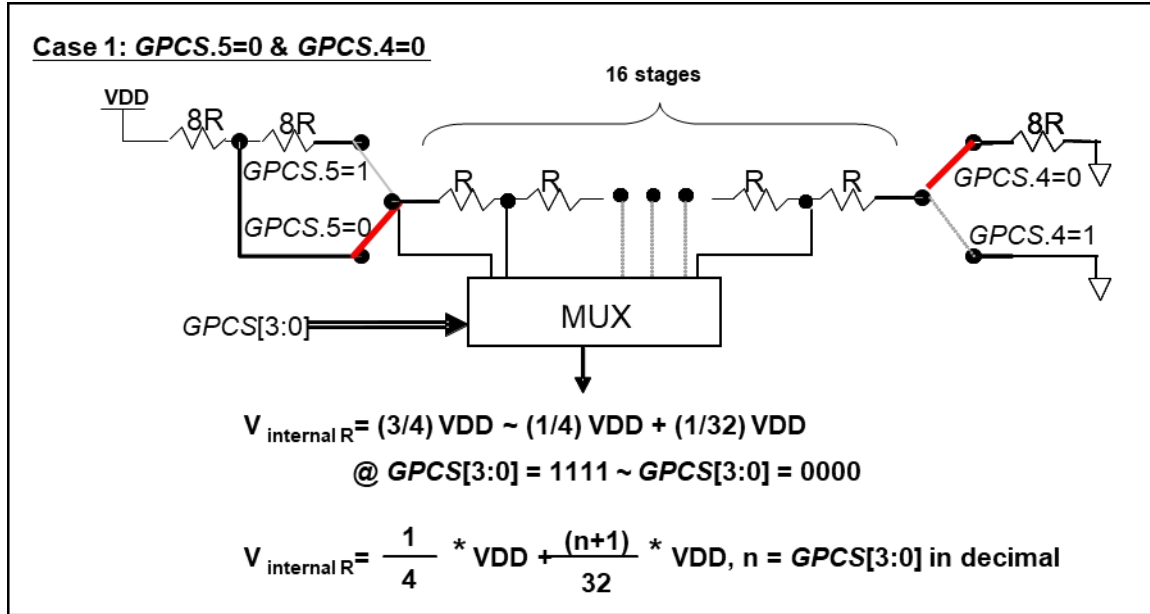


图 14 :  $V_{\text{internal R}}$  硬件接法 ( $\text{GPCS}.5=0$  &  $\text{GPCS}.4=0$ )

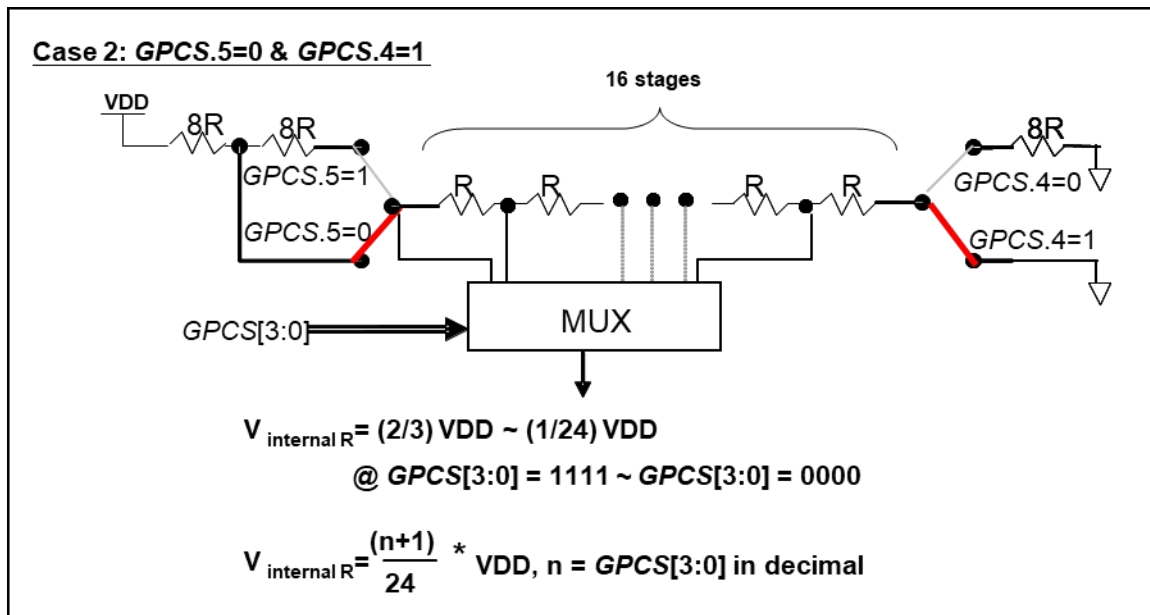


图 15:  $V_{\text{internal R}}$  硬件接法 ( $\text{GPCS}.5=0$  &  $\text{GPCS}.4=1$ )



### Case 3: $GPCS.5=1$ & $GPCS.4=0$

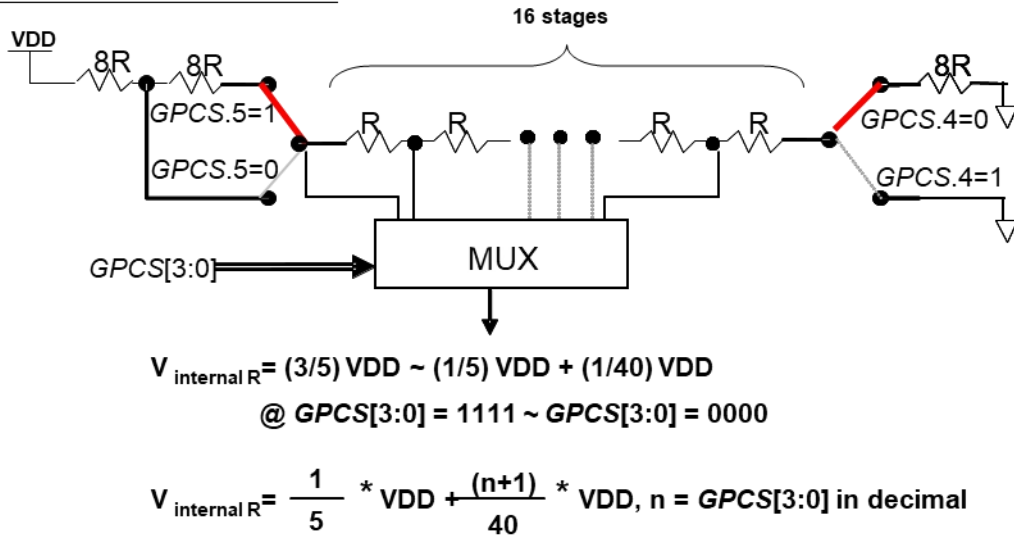


图 16 :  $V_{internal R}$  硬件接法 ( $GPCS.5=1$  &  $GPCS.4=0$ )

### Case 4: $GPCS.5=1$ & $GPCS.4=1$

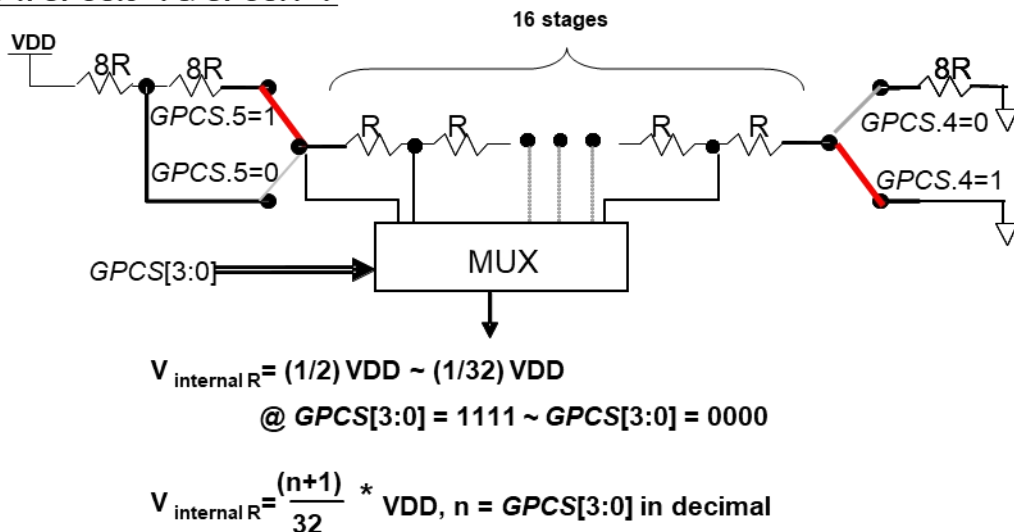


图 17 :  $V_{internal R}$  硬件接法 ( $GPCS.5=1$  &  $GPCS.4=1$ )





### 11.1.4. 使用比较器

例一：

选择PA3 为负输入和 $V_{\text{internal R}}$  的电压为 $(18/32)*V_{\text{DD}}$  作为正输入。 $V_{\text{internal R}}$ 选择上图GPCS[5:4] = 2b'00 的配置方式，GPCS[3:0] = 4b'1001 (n=9) 以得到  $V_{\text{internal R}} = (1/4)*V_{\text{DD}} + [(9+1)/32]*V_{\text{DD}} = [(9+9)/32]*V_{\text{DD}} = (18/32)*V_{\text{DD}}$  的参考电压。

```
GPCS   = 0b0_0_00_1001;      // Vinternal R = VDD*(18/32)
GPCC   = 0b1_0_0_0_000_0;    // 启用比较器，负输入：PA3，正输入：Vinternal R
PADIER = 0bxxxx_0_xxx;      // 停用 PA3 数字输入防止漏电 (x: 由客户自定)
```

或者

```
$ GPCS   VDD*18/32;
$ GPCC   Enable, N_PA3, P_R;  // N_xx 是负输入，P_R 代表正输入是内部参考电压
PADIER = 0bxxxx_0_xxx;
```

例二：

选择  $V_{\text{internal R}}$  为负输入， $V_{\text{internal R}}$  的电压为 $(22/40)*V_{\text{DD}}$ ，选择 PA4 为正输入，比较器的结果反极性并输出到 PA0。 $V_{\text{internal R}}$  选择上图的配置方式 “GPCS[5:4] = 2b'10” 和 GPCS[3:0] = 4b'1101 (n=13) 得到  $V_{\text{internal R}} = (1/5)*V_{\text{DD}} + [(13+1)/40]*V_{\text{DD}} = [(13+9)/40]*V_{\text{DD}} = (22/40)*V_{\text{DD}}$ 。

```
GPCS   = 0b1_0_10_1101;      // 输出到 PA0, Vinternal R = VDD*(22/40)
GPCC   = 0b1_0_0_1_011_1;    // 反极性输出，负输入=Vinternal R，正输入=PA4
PADIER = 0bxxx_0_xxxx;      // 停用 PA4 数字输入防止漏电 (x: 由客户自定)
```

或者

```
$ GPCS   Output, VDD*22/40;
$ GPCC   Enable, Inverse, N_R, P_PA4; // N_R 代表负输入是内部参考电压，P_xx 是正输入
PADIER = 0bxxx_0_xxxx;
```

**注意：**当选择 PA0 做比较器结果输出时，GPCS 会影响 PA3 的仿真输出功能，但不影响实际 IC 的功能，请在仿真时需避开这个情况。



### 11.1.5. 使用比较器和 Bandgap 参考电压生成器

内部 Bandgap 参考电压生成器可以提供 1.20V，它可以测量外部电源电压水平。该 Bandgap 参考电压可以选做负输入去和正输入  $V_{\text{internal R}}$  比较。 $V_{\text{internal R}}$  的电源是 VDD，利用调整  $V_{\text{internal R}}$  电压水平和 Bandgap 参考电压比较，就可以知道 VDD 的电压。

如果 N ( $\text{GPCS}[3:0]$  十进制) 是让  $V_{\text{internal R}}$  最接近 1.20V，那么 VDD 的电压就可以透过下列公式计算：

对于 Case 1 而言： $V_{\text{DD}} = [32 / (N+9)] * 1.20 \text{ volt}$ ；

对于 Case 2 而言： $V_{\text{DD}} = [24 / (N+1)] * 1.20 \text{ volt}$ ；

对于 Case 3 而言： $V_{\text{DD}} = [40 / (N+9)] * 1.20 \text{ volt}$ ；

对于 Case 4 而言： $V_{\text{DD}} = [32 / (N+1)] * 1.20 \text{ volt}$ ；

例一：

```
$ GPCS      VDD*12/40;           // 4.0V * 12/40 = 1.2V
$ GPCC  Enable, BANDGAP, P_R;   // BANDGAP 是负输入, P_R 代表正输入是内部参考电压
....
if (GPC_Out)                     // 或写成 GPCC.6
{
...                               // 当 VDD > 4V
}
else
{
...                               // 当 VDD < 4V
}
```



### 11.2. 触摸功能

JZ8M117 包含一个触摸检测电路，其功能框图如图 18 所示。

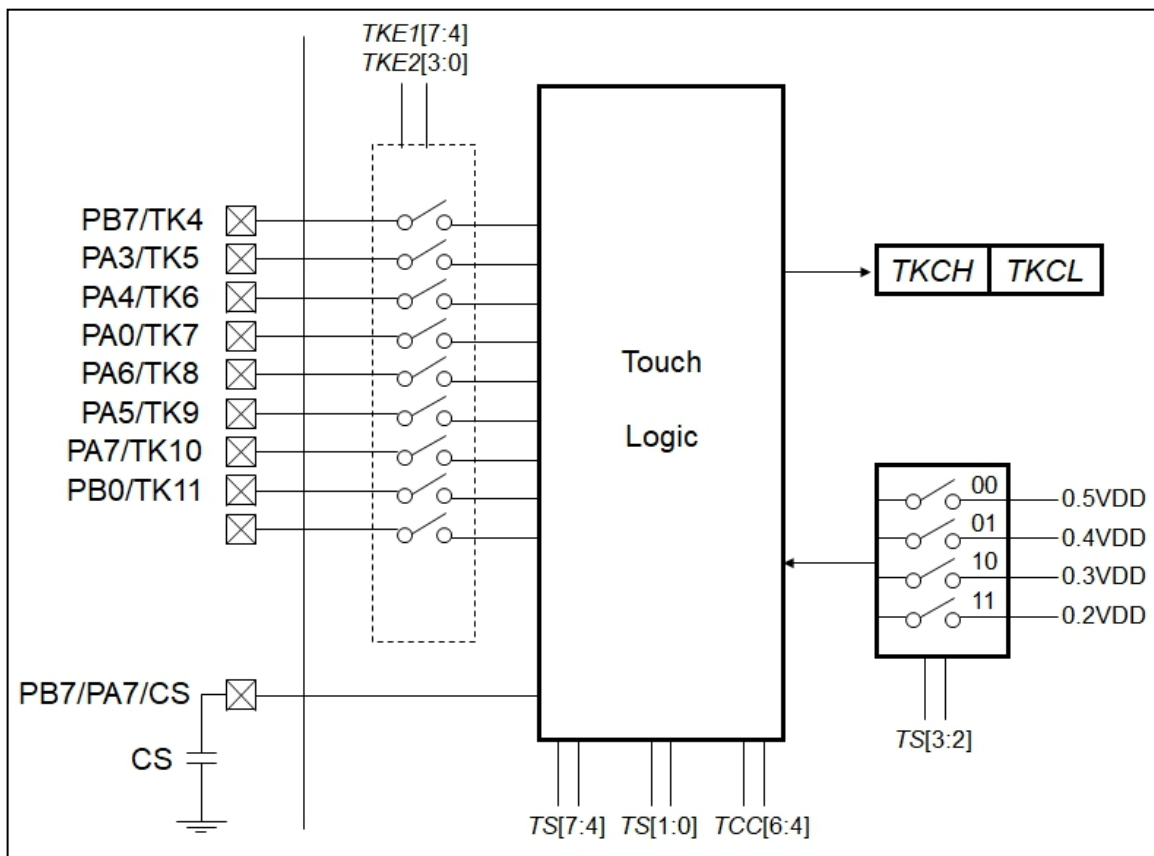


图 18：触摸检测电路功能框图

JZ8M117 中的触摸检测电路采用电容感应的方法，检测手指的虚拟对地电容或感应极片之间的电容。

使用该功能时，需要在 PA7/CS 引脚或 PB7/CS 引脚与 GND 之间连接一颗精确的、X7R 材质的外部电容器 CS。用户可以通过程序选项 CS\_Sel 选择 PA7 或 PB7 中的一个作为 CS 引脚。当 CS 引脚被选中时，另一个引脚可以像其他 IO 一样被选择为触摸按键。

启动触摸检测过程时，用户应遵循以下步骤：

1. 通过设置 *TKE1* 和 *TKE2* 寄存器选择要测量的感应极片（引脚）。每次只能选择一个引脚。
2. 通过在 *TCC* 寄存器中写入“0x10”来发出 *Touch START* 命令。电容 CS 将首先自动放电到 VSS。放电时间可以透过寄存器 *TS[1:0]* 从 32，64 和 128 个触摸电路时钟周期中选择。



## JZ8M117

### 7 触摸键 8 位 MTP 类型单片机

3. 电容值越大，将电容器完全放电到 VSS 所需的放电时间就越长。然而，在某些情况下，128 个触摸时钟可能仍然不够长，不足以将 CS 电容完全放电。此时，用户应该手动将“0x30”写入 TCC 寄存器，而不是“0x10”。在由用户控制的一定放电时间之后，用户可以发出 **Touch START (0x10)**命令来继续此触摸转换进程。或者用户也可以通过将“0x00”写入 TCC 寄存器来中止转换进程。
4. 放电后，CS 会在每个触摸时钟周期（TK\_CLK）朝着 VCC 充电。充电速度由所选感应极片的电容值决定。
5. 当充电过程中的电压达到内部产生的阈值电压(VREF)时，充电过程将自动停止。程序通过读取 TCC[6:4]或 INTRQ[3]来判断充电过程是否停止。VREF 电压可以通过寄存器 TS[3:2]，在  $0.2 \cdot VCC$ 、 $0.3 \cdot VCC$ 、 $0.4 \cdot VCC$  和  $0.5 \cdot VCC$  之间选择。
6. 经过读取触摸计数器 TKCH 和 TKCL 的值，用户可监测感应极片上的电容量变化。读取到的值与 CS 和 CP 的比例有关，而 CP 表示电容可以通过因用户手指的触摸而变化的 PCB，导线和触摸板的组合的总电容。一旦 CP 值被改变，将 CS 充电到 VREF 所需的时间缩短。通过计数时钟周期的差异，电路可以决定触摸板是否启用。

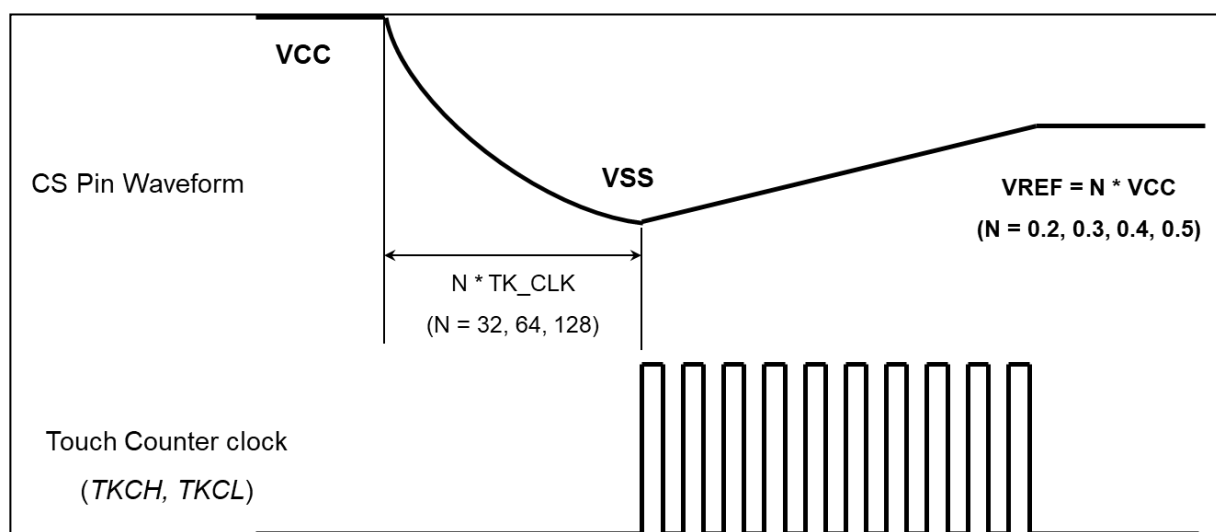


图 19：触摸转换进度时序图

**注意：**当 VREF 电压首次设置或者中途切换参考电压选项时，请舍弃在此之后读取到的第一笔 TKCH 和 TKCL 的数据。



# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

### 11.2.1. 触摸相关寄存器

#### 11.2.1.1. 触摸选项寄存器(TS), 地址 = 0x20

位	初始值	读/写	描述
7 - 4	-	读/写	Touch 时钟(TK_CLK)选择。 0010: 4 MHz 0011: 2 MHz 0100: 1 MHz 0101: 500 kHz 0110: 250 kHz 0111: 125 kHz 1000: ILRC 其他: 保留
3 - 2	00	读/写	Touch VREF 选择 00: 0.5 * VCC 01: 0.4 * VCC 10: 0.3 * VCC 11: 0.2 * VCC
1 - 0	00	读/写	选择触摸功能启动前的放电时间(TK_DISCHG) 00: reserved 01: 32 * CLK 10: 64 * CLK 11: 128 * CLK

#### 11.2.1.2. 触摸充电控制寄存器(TCC), 地址 = 0x21

位	初始值	读/写	描述
7	-	-	保留
6 - 4	-	读/写	触摸控制和状态
			数据      命令(W)      状态(R)
			000      TK_STOP (触摸模块掉电)      准备中/ 结束
			001      TK_RUN (触摸启动指令)      运行中
			011      放电 (CS 电容放电)      放电中
			其他      保留      保留
3 - 0	-	-	保留



# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

### 11.2.1.3. 触摸按键使能 2 寄存器 (TKE2), 地址 = 0x22

位	初始值	读/写	描述
7 - 4	-	-	保留
3	0	读/写	使能 PB0/TK11。0/1: 停用/启用
2	0	读/写	使能 PA7/TK10。0/1: 停用/启用
1	0	读/写	使能 PA5/TK9。0/1: 停用/启用
0	0	读/写	使能 PA6/TK8。0/1: 停用/启用

### 11.2.1.4. 触摸按键使能 1 寄存器 (TKE1), 地址 = 0x24

位	初始值	读/写	描述
7	0	读/写	使能 PA0/TK7。0/1: 停用/启用
6	0	读/写	使能 PA4/TK6。0/1: 停用/启用
5	0	读/写	使能 PA3/TK5。0/1: 停用/启用
4	0	读/写	使能 PB7/TK4。0/1: 停用/启用
3 - 0	-	-	保留

### 11.2.1.5. 触摸按键充电计数高位寄存器(TKCH), 地址= 0x2B

位	初始值	读/写	描述
7 - 4	-	-	保留
3 - 0	-	只读	触摸按键充电计数的 tkct[11:8]

### 11.2.1.6. 触摸按键充电计数低位寄存器(TKCL), 地址= 0x2C

位	初始值	读/写	描述
7 - 0	-	只读	触摸按键充电计数的 tkct[7:0]

### 11.2.1.7. 触摸参数设置寄存器 2(TPS2), 地址= 0x28

位	初始值	读/写	描述
7 - 6	00	R/W	触摸工作模式 00: 模式 A (建议 PCB 上 CS 接 VDD) 01: 模式 B (建议 PCB 上 CS 接 GND) 1X: Reserved
5 - 3	000	R/W	系统保留, 请填 000
2	0	R/W	系统保留, 请填 0
1 - 0	00	R/W	触摸 VREF 续电维持时间选择 00: VREF 续电在 CS 放电完即断开 01: VREF 续电保持。(建议值) 10: VREF 续电在 CS 放电完后继续维持 64 cycles 11: VREF 续电在 CS 放电完后继续维持 128 cycles



## 12. 仿真注意事项

建议使用 PDK5S-I-S01 或 PDK5S-I-S02(B)仿真器。用 PDK5S-I-S01/2(B)仿真时，请注意以下几点：

- (1) 用 PDK5S-I-S01/2(B)仿真时，不支援  $SYSCLK=ILRC/16$  和  $ILRC/2$ 。
- (2) 用 PDK5S-I-S01/2(B)仿真时，不支援 PA5 当中断源。
- (3) 用 PDK5S-I-S01/2(B)仿真时，不支援 GPC\_PWM, TMx\_source, TMx\_bit, CS\_Sel, Interrupt\_Src0, PA3\_PA4\_Drive 等 code option。
- (4) 用 PDK5S-I-S01/2(B)仿真时，不支持 *TM2C.GPCRS* 和 *TM3C.GPCRS*
- (5) 用 PDK5S-I-S01/2(B)仿真时，不支援 *PAPL*、*PBPL*。
- (6) 用 PDK5S-I-S01/2(B)仿真时，不支持所有触摸功能。
- (7) 实际芯片的 *TM2C* PWM 输出是 PA0, PA3 和 PB0，但在仿真器上是 PB2, PA3 和 PB4。
- (8) 实际芯片的 *TM3C* PWM 输出是 PA4, PA5 和 PB7，但在仿真器上是 PB5, PB6 和 PB7。
- (9) 当 *GPCS* 选择 Output 到 PA0 输出时，PA3 的输出功能会受影响。
- (10) 仿真 PWM 波形时，建议用户在程序运行期间查看波形，当仿真器暂停或单步运行时波形可能会与实际不符。
- (11) PDK5S-I-S01/2(B)仿真器的 ILRC 频率与实际 IC 不同，且未经校准，其频率范围大约在 34K~38KHz。
- (12) 快速唤醒时间和使用 PDK5S-I-S01/2(B)仿真不同 (PDK5S-I-S01/2(B): 128 SYSCLK, JZ8M117: 45 ILRC)
- (13) IC 的看门狗溢出时间和使用 PDK5S-I-S01/2(B)仿真不同，如下：

WDT 溢出时间	PDK5S-I-S01/2(B)	JZ8M117
<i>MISC</i> [1:0]=00	$2048 * T_{ILRC}$	$8192 * T_{ILRC}$
<i>MISC</i> [1:0]=01	$4096 * T_{ILRC}$	$16384 * T_{ILRC}$
<i>MISC</i> [1:0]=10	$16384 * T_{ILRC}$	$65536 * T_{ILRC}$
<i>MISC</i> [1:0]=11	$256 * T_{ILRC}$	$262144 * T_{ILRC}$



### 13. 烧录方法

请使用 PDK5S-P-003 进行烧录。PDK3S-P-002 或之前的Writer 版本皆已不支持烧录 JZ8M117。

Jumper 连接：可依照烧录器软件上的说明，连接 jumper 即可。

请用户依据实际情况选择以下两种烧录模式。

#### 13.1. 普通烧录模式

适用范围：

单独封装 IC，并在烧录器的 IC 插座或连接分选机烧录。

合封（MCP）IC，但与 JZ8M117 合封的 IC 及元件不会被以下电压破坏，也不会钳制以下电压的产生。

普通烧录模式电压条件：

- (1) VDD 等于 7.5V，而最大供给电流最高可达约 20mA。
- (2) PA5 等于 6V。
- (3) 其他烧录引脚（GND 除外）等于 VDD。

**重要提示：**

如在 handler 上对 IC 进行烧录，请务必按照 APN004 及 APN011 的指示进行。

为对抗烧录时的杂讯干扰，请于烧录时在分选机连接 IC 连接器一端的 VDD 和 GND 之间连接 0.01uF 电容。

但切忌连接标值 0.01uF 以上的电容，以免影响普通烧录模式的运行。

#### 13.2. 限压烧录模式

适用范围：

在板烧录（On-board Writing），但其周边电路及组件不会被以下电压破坏，也不会钳制以下电压的产生。

请参考在板烧录章节的详细说明。

合封（MCP）IC，但与 JZ8M117 合封的 IC 及元件不会被以下电压破坏，也不会钳制以下电压的产生。

限压烧录模式电压条件：

- (1) VDD 等于 5V，而最大供给电流最高可达约 20mA。
- (2) PA5 等于 5V。
- (3) 其他烧录引脚（GND 除外）等于 VDD。





### 13.3. 在板烧录（On-Board Writing）

JZ8M117 可以支持在板烧录。所谓在板烧录，是指 IC 及其他周边电路及组件，皆已经焊接到 PCB 上，并对 IC 进行烧录的情况。在板烧录需要使用 PDK5S-P-003 上五根引线：ICPCK、ICPDA、VDD、GND 和 ICVPP，用于与 IC 上的 PA3、PA6、VDD、GND 和 PA5 对应相连。

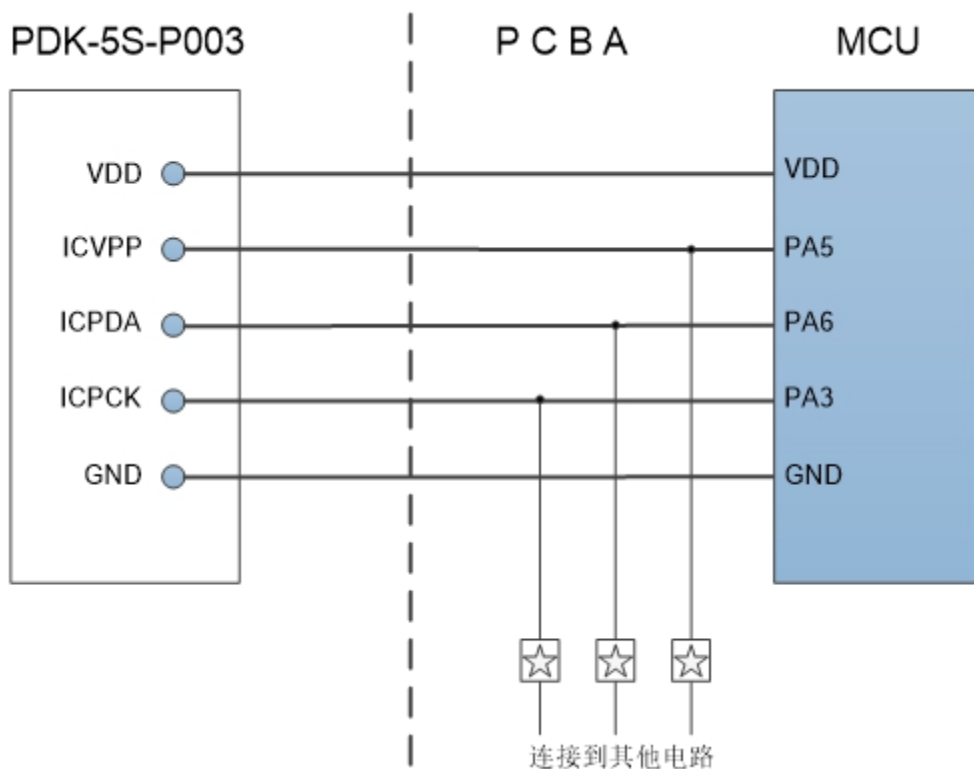


图 20：在板烧录接线示意图

图 20 中的 ☆为电阻或电容，用于隔离烧录接线和其他电路。电阻应 $\geq 10K\Omega$ ，电容应 $\leq 220pF$ 。

#### 注意：

一般来说，在板烧录应使用限压烧录模式。请参考在限压烧录模式的详细说明。

PCB 上的 VDD 与 GND 之间不可连接有 5.0V 或以下的稳压二极管或其他钳制 5.0V 产生的电路或元件。

PCB 上的 VDD 与 GND 之间不可连接有标值 500uF 或以上的电容器。

一般来说，用于烧录讯号的 PA3，PA5 及 PA6 引脚，**不能**作为强输出脚。



## 14. 直流交流电气特性

### 14.1. 绝对最大值

名称	最小值	典型值	最大值	单位	备注
电源电压 (VDD)	2.2		5.5	V	电源电压最大不能超过 <b>5.5V</b> ，否则可能损坏 IC
输入电压	-0.3		$V_{DD} + 0.3$	V	
工作温度	-40		85	°C	
储藏温度	-50		125	°C	
节点温度		150		°C	

### 14.2. 器件电气特性

下列所有数据除特别列明外，皆于  $V_{DD}=3.3V$ ， $f_{SYS}=2MHz$  之条件下获得。

符号	特性	最小值	典型值	最大值	单位	条件( $T_a=25^{\circ}C$ )
$V_{DD}$	工作电压	2.2*	5.0	5.5	V	* 受限于 LVR 公差
LVR%	低电压复位公差	-5		5	%	
$f_{SYS}$	系统时钟(CLK)* = IHRC/2 IHRC/4 IHRC/8 ILRC	0 0 0	64K	8M 4M 2M	Hz	$V_{DD} \geq 3.5V$ $V_{DD} \geq 2.5V$ $V_{DD} \geq 2.2V$ $V_{DD} = 5.0V$
$P_{cycle}$	烧录次数	1000			cycles	
$I_{OP}$	工作电流		0.7 82		mA uA	$f_{SYS}=IHRC/16=1MIPS@5.0V$ $f_{SYS}=ILRC=64KHz@5.0V$
$I_{PD}$	掉电模式消耗电流 (使用 <i>stopsys</i> 命令)		1		uA uA	$V_{DD}=5.0V$
$I_{PS}$	省电模式消耗电流 (使用 <i>stopexe</i> 命令)		3		uA	$V_{DD}=5.0V$ ; $f_{SYS}=ILRC$ 仅使用 ILRC 的模式下
$V_{IL}$	输入低电压	0		$0.1 V_{DD}$	V	
$V_{IH}$	输入高电压	$0.7 V_{DD}$		$V_{DD}$	V	
$I_{OL}$	IO 输出灌电流 (Strong) PA3/PA4 Others IO 输出灌电流 (Normal)		40 23 23		mA	$V_{DD}=5.0V$ , $V_{OL}=0.5V$
$I_{OH}$	IO 输出驱动电流 (Strong) PA3/PA4 Others IO 输出驱动电流 (Normal)		20 10 10		mA	$V_{DD}=5.0V$ , $V_{OH}=4.5V$



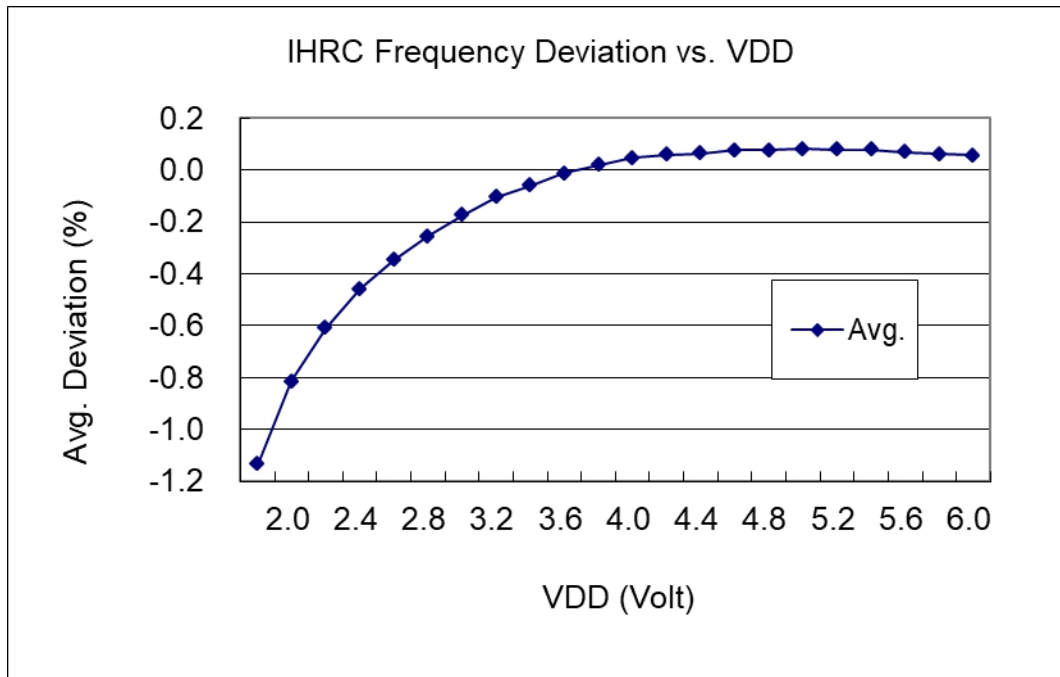
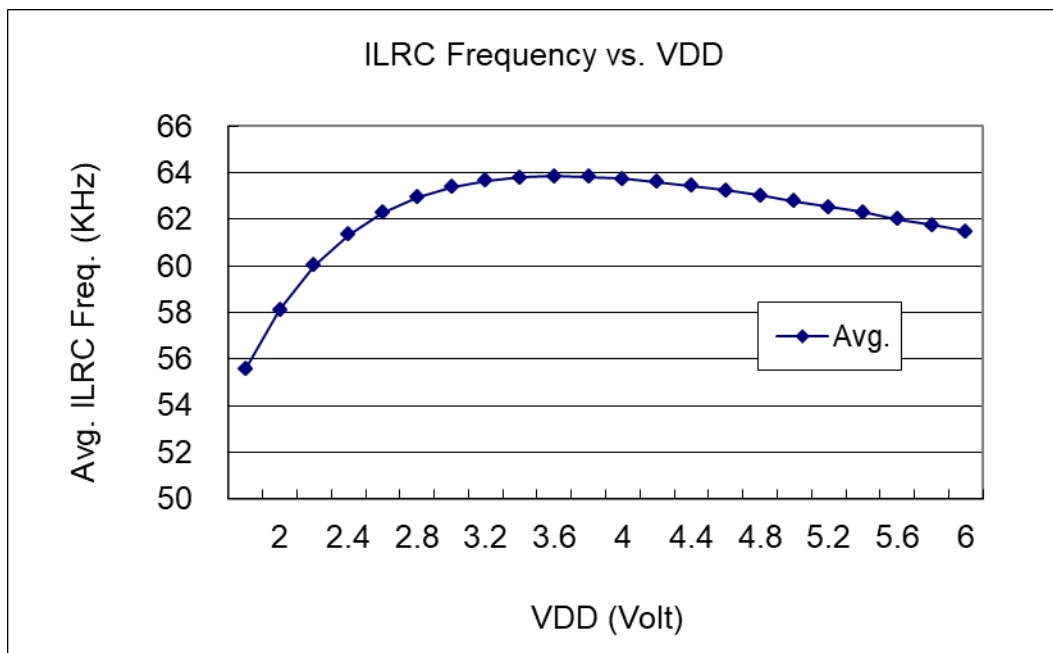
# JZ8M117

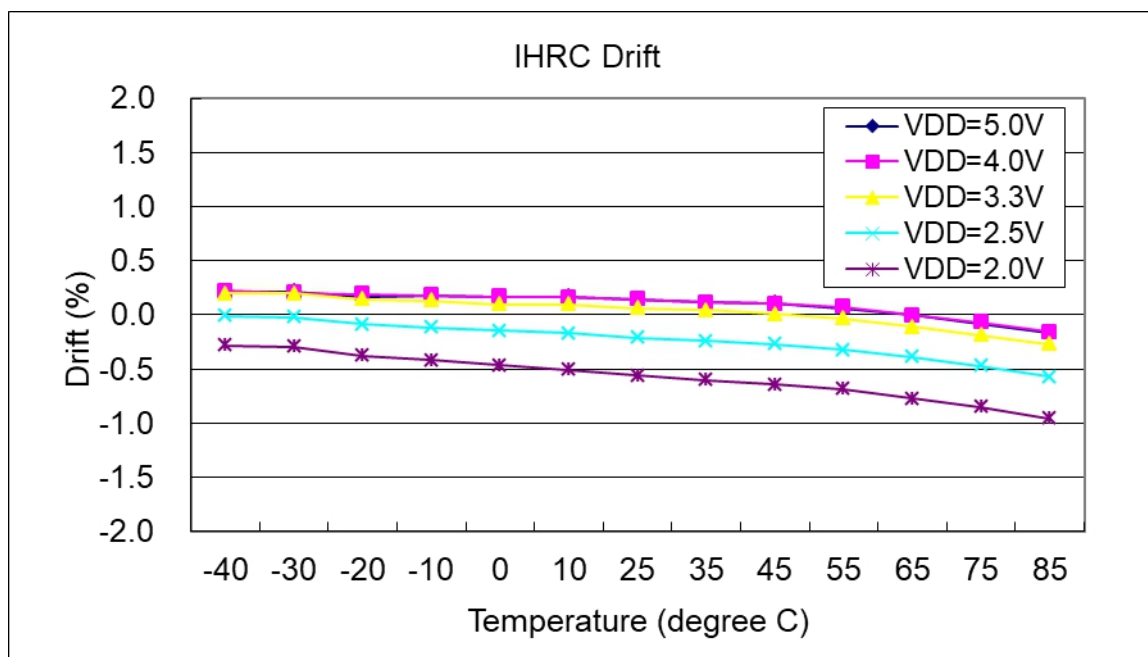
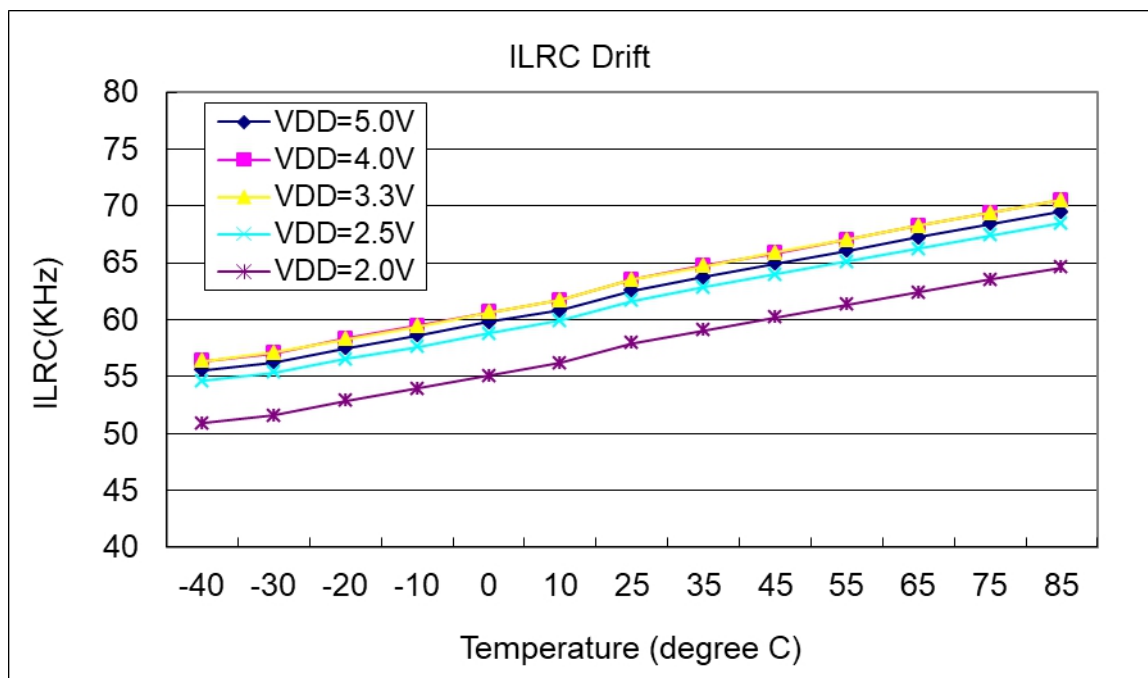
## 7 触摸键 8 位 MTP 类型单片机

符号	特性	最小值	典型值	最大值	单位	条件(Ta=25°C)
V <sub>IN</sub>	输入电压	-0.3		V <sub>DD</sub> +0.3	V	
I <sub>INJ</sub> (PIN)	脚位的引入电流		1		uA	V <sub>DD</sub> +0.3 ≥ V <sub>IN</sub> ≥ -0.3
R <sub>PH</sub>	上拉电阻 PA5 其他		85 80		KΩ	V <sub>DD</sub> =5.0V
R <sub>PL</sub>	下拉电阻 PA5 其他		85 80		KΩ	V <sub>DD</sub> =5.0V
f <sub>IHRC</sub>	IHRC 输出频率（校准后）*	15.84*	16*	16.16*	MHz	V <sub>DD</sub> =5V, Ta=25 °C
		15.20*	16*	16.80*		V <sub>DD</sub> =2.2V~5.5V, -40 °C <Ta<85 °C *
t <sub>INT</sub>	中断脉冲宽度	30			ns	V <sub>DD</sub> = 5.0V
t <sub>WDT</sub>	看门狗超时溢出时间		8k		T <sub>ILRC</sub>	misc[1:0]=00（默认）
			16k			misc[1:0]=01
			64k			misc[1:0]=10
			256k			misc[1:0]=11
t <sub>SBP</sub>	系统上电开机时间		50		ms	@ V <sub>DD</sub> =5V
t <sub>RST</sub>	外部复位脉冲宽度	120			us	@ V <sub>DD</sub> =5V
CP <sub>os</sub>	比较器偏压*	-	±10	±20	mV	
CP <sub>cm</sub>	比较器共模输入电压*	0		V <sub>DD</sub> -1.5	V	
CP <sub>spt</sub>	比较器响应时间*		100	500	ns	上升沿和下降沿一样
CP <sub>mc</sub>	比较器模式改变稳定时间		2.5	7.5	us	
CP <sub>cs</sub>	比较器电流消耗		20		uA	V <sub>DD</sub> = 3.3V

\*这些参数是设计参考值，并不是每个芯片测试。

特性图是实际测量值。考虑到生产飘移等因素的影响，表格中的数据是在实际测量值的安全范围内。

**14.3. IHRC 频率与 VDD 关系曲线图（校准到 16MHz）****14.4. ILRC 频率与 VDD 关系曲线图**

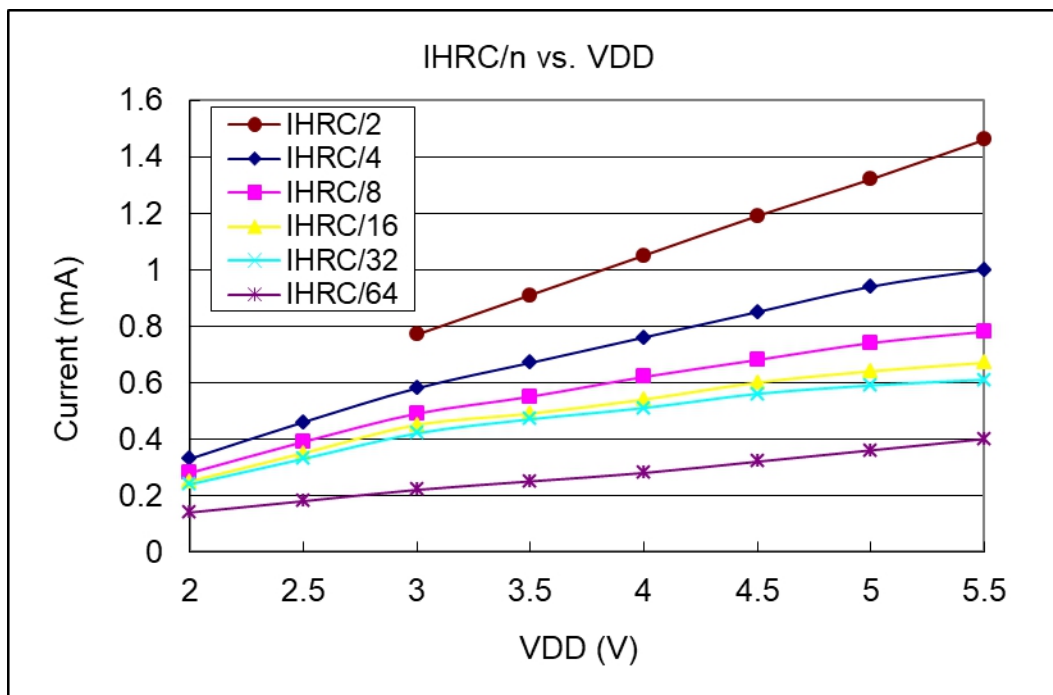
**14.5. IHRC 频率与温度关系曲线图（校准到 16MHz）****14.6. ILRC 频率与温度关系曲线图**



#### 14.7. 工作电流与 VDD、系统时钟 CLK = IHRC/n 关系曲线图

条件: 开启: IHRC, Bandgap, LVR; 关闭: T16, IHRC, Touch module;

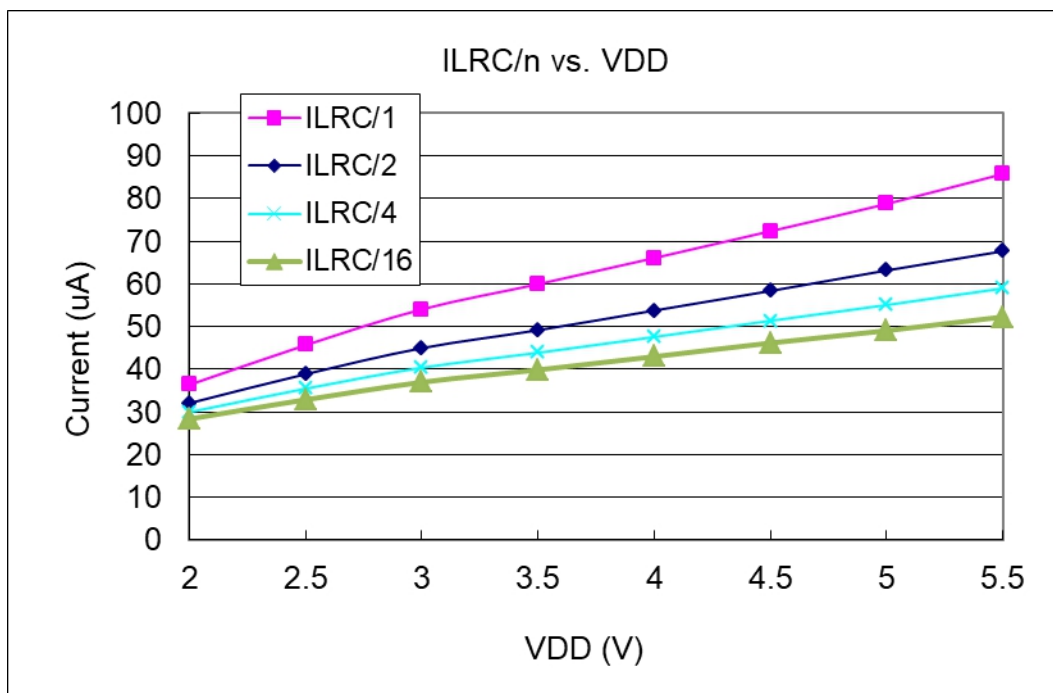
IO: PA0 以 0.5Hz 频率高低电压交换输出, 无负载; 其他: 设为输入且不浮空



#### 14.8. 工作电流与 VDD、系统时钟 CLK = ILRC/n 关系曲线图

条件: 开启: ILRC, Bandgap, LVR; 关闭: T16, IHRC, Touch module;

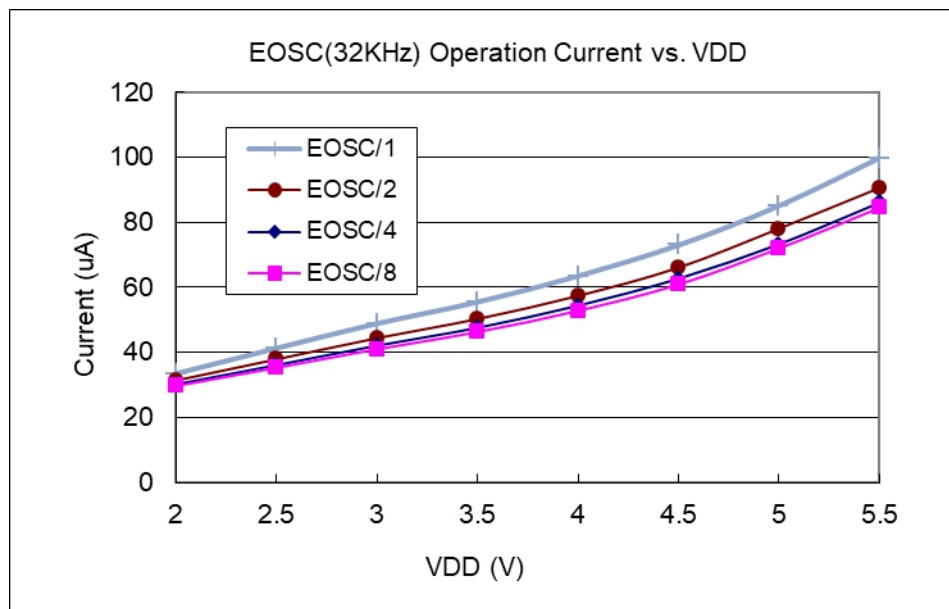
IO: PA0 以 0.5Hz 频率高低电压交换输出, 无负载; 其他: 设为输入且不浮空



**14.9. 工作电流与VDD、系统时钟 CLK = 32KHz EOSC / n 关系曲线图**

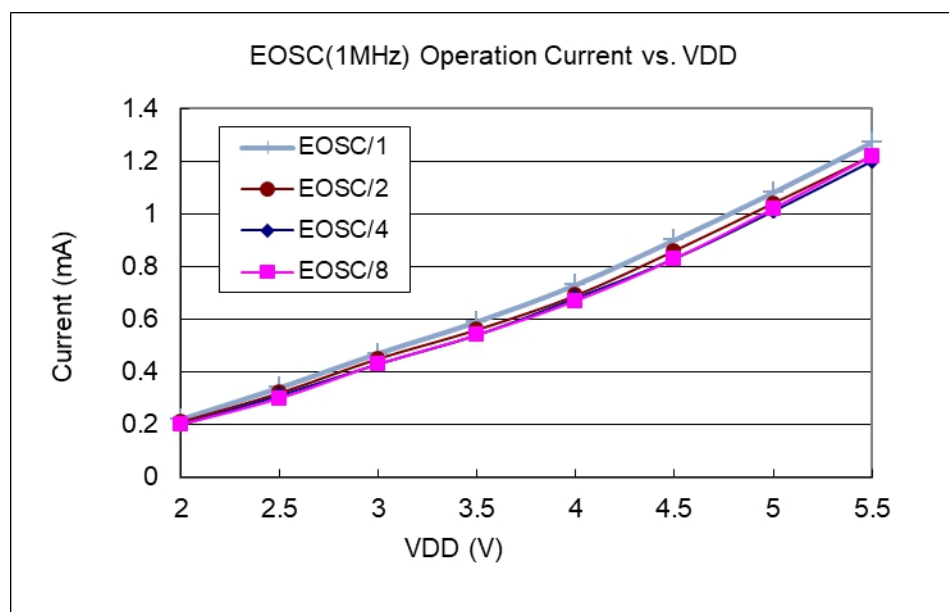
条件: 开启: EOSC[6,5] = [0,1], Bandgap, LVR; 关闭: T16, IHRC, Touch module;

IO: PA0 以 0.5Hz 频率高低电压交换输出, 无负载; 其他: 设为输入且不浮空

**14.10. 工作电流与 VDD、系统时钟 CLK = 1MHz EOSC / n 关系曲线图**

条件: 开启: EOSC[6,5] = [1,0], Bandgap, LVR; 关闭: T16, IHRC, Touch module;

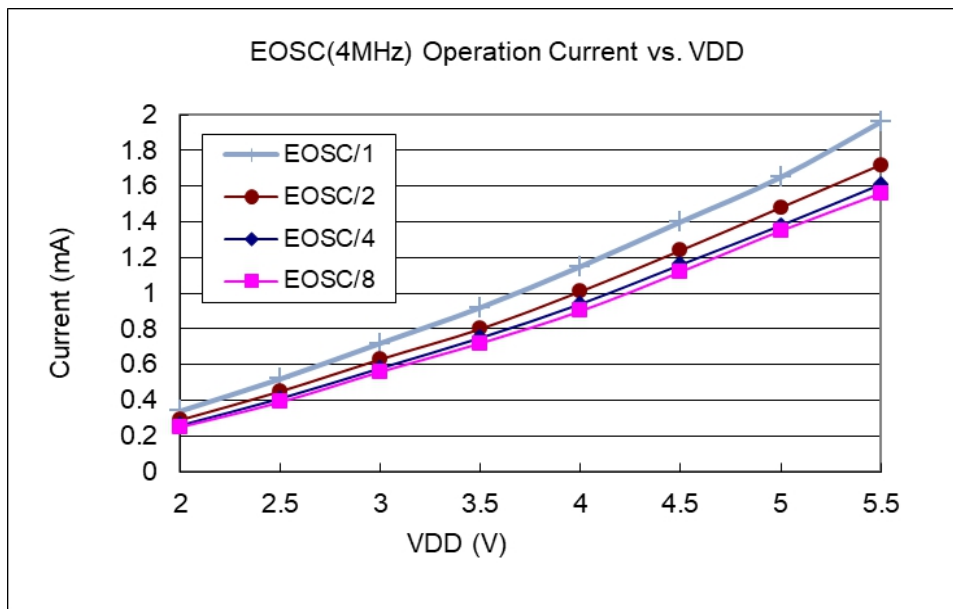
IO: PA0 以 0.5Hz 频率高低电压交换输出, 无负载; 其他: 设为输入且不浮空



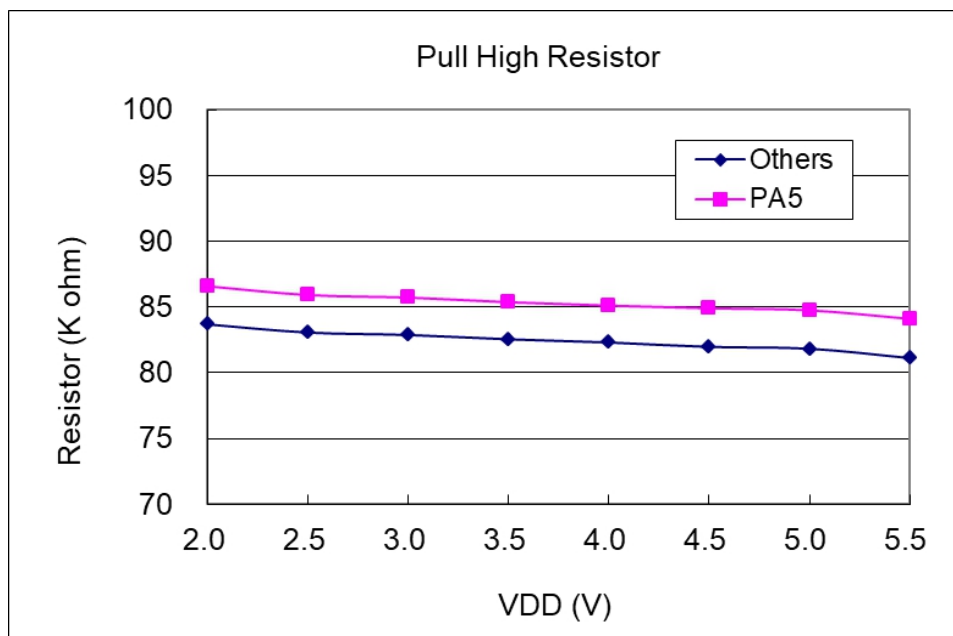


### 14.11. 工作电流与 VDD、系统时钟 CLK = 4MHz EOSC / n 关系曲线图

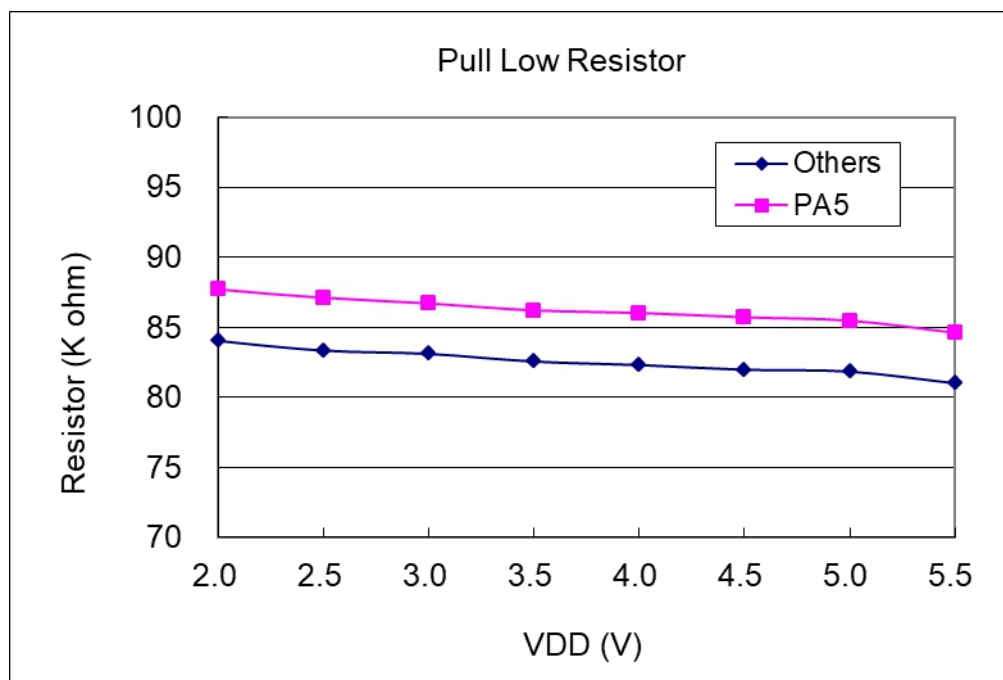
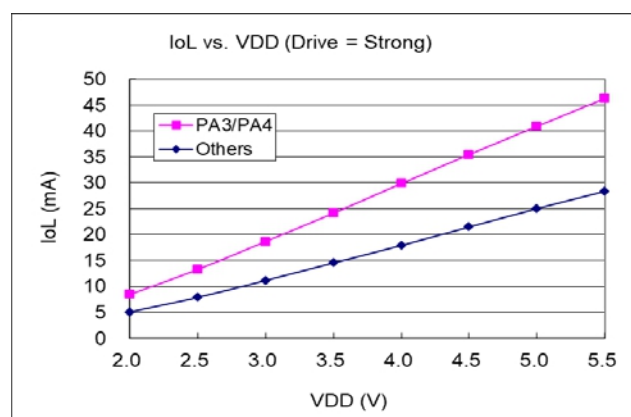
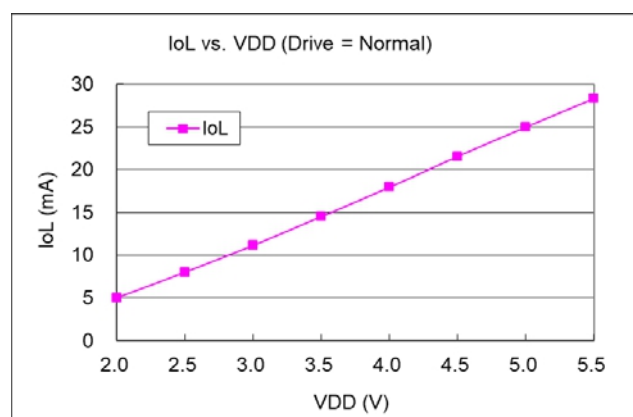
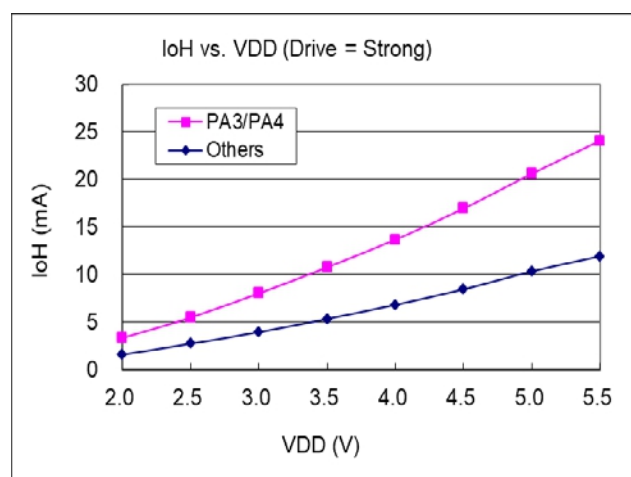
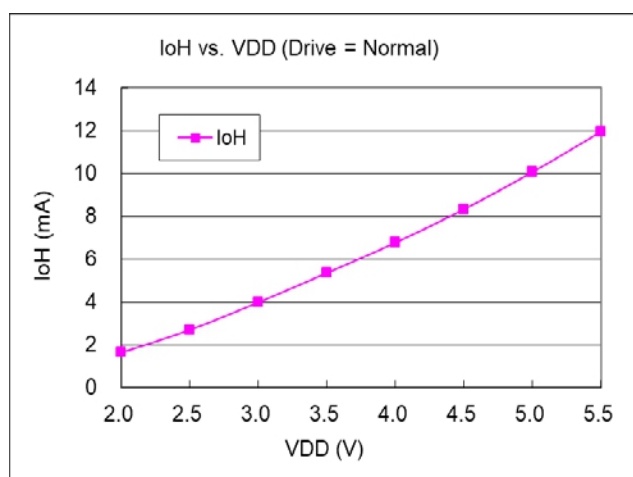
条件: 开启: EOSC[6,5] = [1,1], Bandgap, LVR; 关闭: T16, IHRC, Touch module;  
IO: PA0 以 0.5Hz 频率高低电压交换输出, 无负载; 其他: 设为输入且不浮空

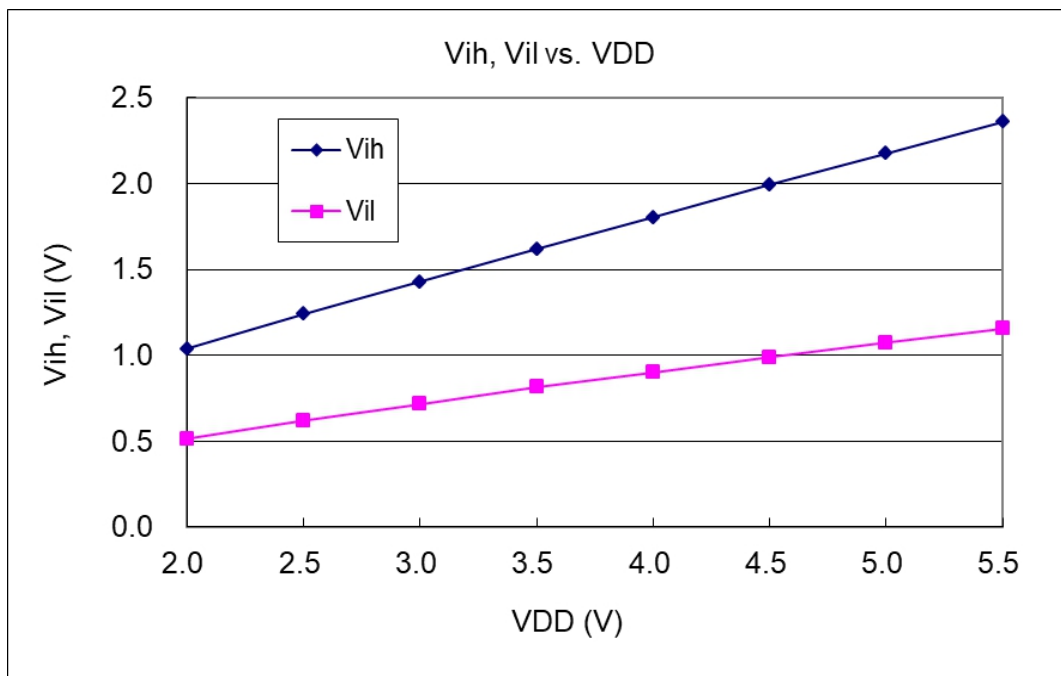
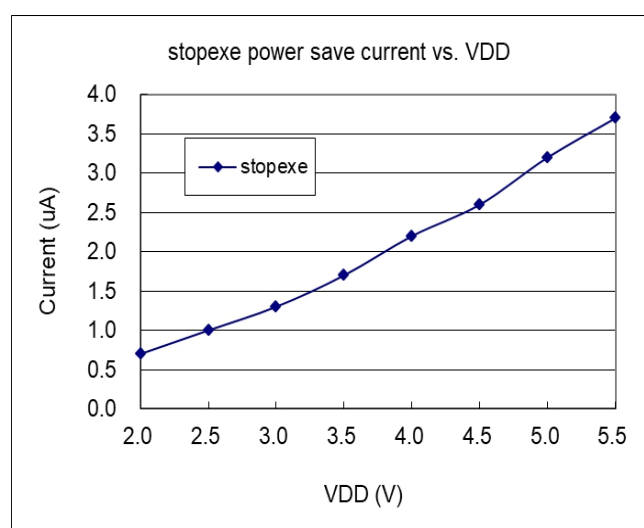
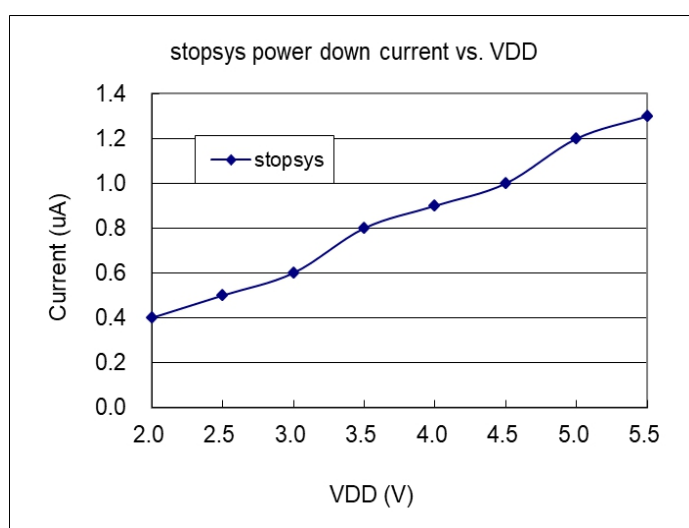


### 14.12. IO 引脚上拉阻抗曲线图





**14.13. IO 引脚下拉阻抗曲线图****14.14. IO 引脚输出驱动电流( $I_{OH}$ ) 和灌电流 ( $I_{OL}$ )曲线图** $(V_{OH}=0.9*V_{DD}, V_{OL}=0.1*V_{DD})$ 

**14.15. IO 引脚输入高/低阈值电压( $V_{IH}/V_{IL}$ )曲线图****14.16. 掉电模式消耗电流( $I_{PD}$ )与省电模式消耗电流( $I_{PS}$ )曲线图**



## 15. 指令

符 号	描 述
<b>ACC</b>	累加器 (Accumulator 的缩写)
<b>a</b>	累加器 (Accumulator 在程序里的代表符号)
<b>SP</b>	堆栈指针
<b>FLAG</b>	标志寄存器
<b>I</b>	即时数据
<b>&amp;</b>	逻辑 AND
<b> </b>	逻辑 OR
<b>←</b>	移动
<b>^</b>	异或 OR
<b>+</b>	加
<b>—</b>	减
<b>~</b>	NOT (逻辑补数, 1 补数)
<b>⌋</b>	2 补数
<b>OV</b>	溢出 (2 补数系统的运算结果超出范围)
<b>Z</b>	零 (如果零运算单元操作的结果是 0, 这位设置为 1)
<b>C</b>	进位(Carry)
<b>AC</b>	辅助进位标志(Auxiliary Carry)。
<b>IO.n</b>	寄存器的位
<b>M.n</b>	只允许寻址在 address 0~0x3F (0~63) 的位置



### 15.1. 指令表

指令	功能	周期	Z	C	AC	OV
<b>数据传输类指令</b>						
<i>mov</i> a, l	<i>mov</i> a, 0x0f; $a \leftarrow 0fh$ ;	1	-	-	-	-
<i>mov</i> M, a	<i>mov</i> MEM, a; $MEM \leftarrow a$	1	-	-	-	-
<i>mov</i> a, M	<i>mov</i> a, MEM; $a \leftarrow MEM$ ; 当 MEM 为零时, 标志位 Z 会被置位	1	Y	-	-	-
<i>mov</i> a, IO	<i>mov</i> a, pa; $a \leftarrow pa$ ; 当 pa 为零时, 标志位 Z 会被置位	1	Y	-	-	-
<i>mov</i> IO, a	<i>mov</i> pb, a; $pb \leftarrow a$ ;	1	-	-	-	-
<i>ldt16</i> word	<i>ldt16</i> word; $word \leftarrow 16\text{-bit timer}$	1	-	-	-	-
<i>stt16</i> word	<i>stt16</i> word; $16\text{-bit timer} \leftarrow word$	1	-	-	-	-
<i>idxm</i> a, index	<i>idxm</i> a, index; $a \leftarrow [index]$ , index 是以 word 定义	2	-	-	-	-
<i>idxm</i> index, a	<i>idxm</i> index, a; $[index] \leftarrow a$ ; index 是以 word 定义	2	-	-	-	-
<i>xch</i> M	<i>xch</i> MEM; $MEM \leftarrow a, a \leftarrow MEM$	1	-	-	-	-
<i>pushaf</i>	<i>pushaf</i> ; $[sp] \leftarrow \{flag, ACC\}$ ; $sp \leftarrow sp + 2$ ;	1	-	-	-	-
<i>popaf</i>	<i>popaf</i> ; $sp \leftarrow sp - 2$ ; $\{Flag, ACC\} \leftarrow [sp]$ ;	1	Y	Y	Y	Y
<b>算术运算类指令</b>						
<i>add</i> a, l	<i>add</i> a, 0x0f; $a \leftarrow a + 0fh$	1	Y	Y	Y	Y
<i>add</i> a, M	<i>add</i> a, MEM; $a \leftarrow a + MEM$	1	Y	Y	Y	Y
<i>add</i> M, a	<i>add</i> MEM, a; $MEM \leftarrow a + MEM$	1	Y	Y	Y	Y
<i>addc</i> a, M	<i>addc</i> a, MEM; $a \leftarrow a + MEM + C$	1	Y	Y	Y	Y
<i>addc</i> M, a	<i>addc</i> MEM, a; $MEM \leftarrow a + MEM + C$	1	Y	Y	Y	Y
<i>addc</i> a	<i>addc</i> a; $a \leftarrow a + C$	1	Y	Y	Y	Y
<i>addc</i> M	<i>addc</i> MEM; $MEM \leftarrow MEM + C$	1	Y	Y	Y	Y
<i>nadd</i> a, M	<i>nadd</i> a, MEM; $a \leftarrow \neg a + MEM$	1	Y	Y	Y	Y
<i>nadd</i> M, a	<i>nadd</i> MEM, a; $MEM \leftarrow \neg MEM + a$	1	Y	Y	Y	Y
<i>sub</i> a, l	<i>sub</i> a, 0x0f; $a \leftarrow a - 0fh$ ( $a + [2\text{'s complement of } 0fh]$ )	1	Y	Y	Y	Y
<i>sub</i> a, M	<i>sub</i> a, MEM; $a \leftarrow a - MEM$ ( $a + [2\text{'s complement of } M]$ )	1	Y	Y	Y	Y
<i>sub</i> M, a	<i>sub</i> MEM, a; $MEM \leftarrow MEM - a$ ( $MEM + [2\text{'s complement of } a]$ )	1	Y	Y	Y	Y
<i>subc</i> a, M	<i>subc</i> MEM, a; $a \leftarrow a - MEM - C$	1	Y	Y	Y	Y
<i>subc</i> M, a	<i>subc</i> MEM, a; $MEM \leftarrow MEM - a - C$	1	Y	Y	Y	Y
<i>subc</i> a	<i>subc</i> a; $a \leftarrow a - C$	1	Y	Y	Y	Y
<i>subc</i> M	<i>subc</i> MEM; $MEM \leftarrow MEM - C$	1	Y	Y	Y	Y
<i>inc</i> M	<i>inc</i> MEM; $MEM \leftarrow MEM + 1$	1	Y	Y	Y	Y
<i>dec</i> M	<i>dec</i> MEM; $MEM \leftarrow MEM - 1$	1	Y	Y	Y	Y
<i>clear</i> M	<i>clear</i> MEM; $MEM \leftarrow 0$	1	-	-	-	-



指令	功能	周期	Z	C	AC	OV
移位运算类指令						
<i>sr a</i>	<i>sr a</i> ; $a(0, b7, b6, b5, b4, b3, b2, b1) \leftarrow a(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow a(b0)$	1	-	Y	-	-
<i>src a</i>	<i>src a</i> ; $a(c, b7, b6, b5, b4, b3, b2, b1) \leftarrow a(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow a(b0)$	1	-	Y	-	-
<i>sr M</i>	<i>sr MEM</i> ; $MEM(0, b7, b6, b5, b4, b3, b2, b1) \leftarrow MEM(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow MEM(b0)$	1	-	Y	-	-
<i>src M</i>	<i>src MEM</i> ; $MEM(c, b7, b6, b5, b4, b3, b2, b1) \leftarrow MEM(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow MEM(b0)$	1	-	Y	-	-
<i>sl a</i>	<i>sl a</i> ; $a(b6, b5, b4, b3, b2, b1, b0, 0) \leftarrow a(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow a(b7)$	1	-	Y	-	-
<i>slc a</i>	<i>slc a</i> ; $a(b6, b5, b4, b3, b2, b1, b0, c) \leftarrow a(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow a(b7)$	1	-	Y	-	-
<i>sl M</i>	<i>sl MEM</i> ; $MEM(b6, b5, b4, b3, b2, b1, b0, 0) \leftarrow MEM(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow MEM(b7)$	1	-	Y	-	-
<i>slc M</i>	<i>slc MEM</i> ; $MEM(b6, b5, b4, b3, b2, b1, b0, C) \leftarrow MEM(b7, b6, b5, b4, b3, b2, b1, b0)$ , $C \leftarrow MEM(b7)$	1	-	Y	-	-
<i>swap a</i>	<i>swap a</i> ; $a(b3, b2, b1, b0, b7, b6, b5, b4) \leftarrow a(b7, b6, b5, b4, b3, b2, b1, b0)$	1	-	-	-	-
逻辑运算类指令						
<i>and a, l</i>	<i>and a, 0x0f</i> ; $a \leftarrow a \& 0fh$	1	Y	-	-	-
<i>and a, M</i>	<i>and a, RAM10</i> ; $a \leftarrow a \& RAM10$	1	Y	-	-	-
<i>and M, a</i>	<i>and MEM, a</i> ; $MEM \leftarrow a \& MEM$	1	Y	-	-	-
<i>or a, l</i>	<i>or a, 0x0f</i> ; $a \leftarrow a   0fh$	1	Y	-	-	-
<i>or a, M</i>	<i>or a, MEM</i> ; $a \leftarrow a   MEM$	1	Y	-	-	-
<i>or M, a</i>	<i>or MEM, a</i> ; $MEM \leftarrow a   MEM$	1	Y	-	-	-
<i>xor a, l</i>	<i>xor a, 0x0f</i> ; $a \leftarrow a \wedge 0fh$	1	Y	-	-	-
<i>xor IO, a</i>	<i>xor pa, a</i> ; $pa \leftarrow a \wedge pa$ ;	1	-	-	-	-
<i>xor a, M</i>	<i>xor a, MEM</i> ; $a \leftarrow a \wedge RAM10$	1	Y	-	-	-
<i>xor M, a</i>	<i>xor MEM, a</i> ; $MEM \leftarrow a \wedge MEM$	1	Y	-	-	-
<i>not a</i>	<i>not a</i> ; $a \leftarrow \sim a$	1	Y	-	-	-
<i>not M</i>	<i>not MEM</i> ; $MEM \leftarrow \sim MEM$	1	Y	-	-	-
<i>neg a</i>	<i>neg a</i> ; $a \leftarrow a$ 的 2 补码	1	Y	-	-	-
<i>neg M</i>	<i>neg MEM</i> ; $MEM \leftarrow MEM$ 的 2 补码	1	Y	-	-	-
<i>comp a, M</i>	<i>comp a, MEM</i> ; 等效于 $(a - MEM)$ , 并改变标志位 Flag	1	Y	Y	Y	Y
<i>comp M, a</i>	<i>comp MEM, a</i> ; 等效于 $(MEM - a)$ , 并改变标志位 Flag	1	Y	Y	Y	Y



# JZ8M117

## 7 触摸键 8 位 MTP 类型单片机

指令	功能	周期	Z	C	AC	OV
<b>位运算类指令</b>						
<i>set0</i> IO.n	<i>set0</i> pa.5 ; PA5=0	1	-	-	-	-
<i>set1</i> IO.n	<i>set1</i> pb.5 ; PB5=1	1	-	-	-	-
<i>set0</i> M.n	<i>set0</i> MEM.5 ; MEM 位 5 为 0	1	-	-	-	-
<i>set1</i> M.n	<i>set1</i> MEM.5 ; MEM 位 5 为 1	1	-	-	-	-
<i>swapc</i> IO.n	<i>swapc</i> IO.0; C $\leftarrow$ IO.0 , IO.0 $\leftarrow$ C 当 IO.0 是输出脚位, 进位标志 C 将被送到 IO.0 脚 当 IO.0 是输入脚位, IO.0 脚的状态将被送到进位标志 C	1	-	Y	-	-
<b>条件运算类指令</b>						
<i>ceqsn</i> a, l	<i>ceqsn</i> a, 0x55; <i>inc</i> MEM; <i>goto</i> error ; 假如 a=0x55, then “goto error”; 否则, “inc MEM”.	1 / 2	Y	Y	Y	Y
<i>ceqsn</i> a, M	<i>ceqsn</i> a, MEM; 假如 a=MEM, 跳过下一个指令	1 / 2	Y	Y	Y	Y
<i>cneqsn</i> a, M	<i>cneqsn</i> a, MEM; 假如 a $\neq$ MEM, 跳过下一个指令	1 / 2	Y	Y	Y	Y
<i>cneqsn</i> a, l	<i>cneqsn</i> a, 0x55; <i>inc</i> MEM; <i>goto</i> error ; 假如 a $\neq$ 0x55, then “goto error”; 否则, “inc MEM”.	1 / 2	Y	Y	Y	Y
<i>t0sn</i> IO.n	<i>t0sn</i> pa.5; 如果 PA5 是 0, 跳过下一个指令	1 / 2	-	-	-	-
<i>t1sn</i> IO.n	<i>t1sn</i> pa.5; 如果 PA5 是 1, 跳过下一个指令	1 / 2	-	-	-	-
<i>t0sn</i> M.n	<i>t0sn</i> MEM.5; 如果 MEM 的位 5 是 0, 跳过下一个指令	1 / 2	-	-	-	-
<i>t1sn</i> M.n	<i>t1sn</i> MEM.5; 如果 MEM 的位 5 是 1, 跳过下一个指令	1 / 2	-	-	-	-
<i>izsn</i> a	<i>izsn</i> a; a $\leftarrow$ a + 1, 若 a=0, 跳过下一个指令	1 / 2	Y	Y	Y	Y
<i>dzsn</i> a	<i>dzsn</i> a; a $\leftarrow$ a - 1, 若 a=0, 跳过下一个指令	1 / 2	Y	Y	Y	Y
<i>izsn</i> M	<i>izsn</i> MEM; MEM $\leftarrow$ MEM + 1, 若 MEM=0, 跳过下一个指令	1 / 2	Y	Y	Y	Y
<i>dzsn</i> M	<i>dzsn</i> MEM; MEM $\leftarrow$ MEM - 1, 若 MEM=0, 跳过下一个指令	1 / 2	Y	Y	Y	Y
<b>系统控制类指令</b>						
<i>call</i> label	<i>call</i> function1; [sp] $\leftarrow$ pc + 1, pc $\leftarrow$ function1, sp $\leftarrow$ sp + 2	2	-	-	-	-
<i>goto</i> label	<i>goto</i> error; 跳到 error 并继续执行程序	2	-	-	-	-
<i>ret</i> l	<i>ret</i> 0x55; A $\leftarrow$ 55h <i>ret</i> ;	2	-	-	-	-
<i>ret</i>	<i>ret</i> ; sp $\leftarrow$ sp - 2 pc $\leftarrow$ [sp]	2	-	-	-	-
<i>reti</i>	<i>reti</i> ; 从中断服务程序返回到原程序。 在这指令执行之后, 全部中断将自动启用	2	-	-	-	-
<i>nop</i>	<i>nop</i> ; 没有任何改变	1	-	-	-	-
<i>pcadd</i> a	<i>pcadd</i> a; pc $\leftarrow$ pc + a	2	-	-	-	-
<i>engint</i>	<i>engint</i> ; 中断要求可送到 FPP0, 以便进行中断服务	1	-	-	-	-
<i>disgint</i>	<i>disgint</i> ; 送到 FPP0 的中断要求全部被挡住, 无法进行中断服务	1	-	-	-	-
<i>stopsys</i>	<i>stopsys</i> ; 停止系统时钟和关闭系统	1	-	-	-	-
<i>stopexe</i>	<i>stopexe</i> ; 停住系统时钟, 但是仍保持震荡器模块工作	1	-	-	-	-
<i>reset</i>	<i>reset</i> ; 复位整个单片机	1	-	-	-	-
<i>wdreset</i>	<i>wdreset</i> ; 复位看门狗	1	-	-	-	-