

FT62F08X

Application note

目录

1. 12BIT ADC 模块.....	3
1.1 ADC 的配置.....	3
1.1.1 校准 ADC	4
1.1.2 端口配置	4
1.1.3 通道选择	4
1.1.4 触发方式选择	4
1.1.5 触发源选择	5
1.1.6 触发类型选择	5
1.1.7 触发延时配置	5
1.1.8 ADC 参考电压	5
1.1.9 转换时钟	5
1.1.10 中断	7
1.1.11 转换结果的格式	7
1.1.12 阈值比较	7
1.2 ADC 的工作原理.....	8
1.2.1 启动自动校准	8
1.2.2 启动转换	8
1.2.3 转换完成	8
1.2.4 终止转换	9
1.2.5 休眠模式下 ADC 的工作	9
1.2.6 外部触发器	9
1.2.7 A/D 转换步骤	10
1.3 A/D 采集时间要求	11
1.4 与 ADC 相关寄存器汇总.....	12
1.4.1 ADRESL, 地址 0x9B	13
1.4.2 ADRESH, 地址 0x9C	13
1.4.3 ADCON0, 地址 0x9D	13
1.4.4 ADCON1, 地址 0x9E.....	14
1.4.5 ADCON2, 地址 0x9F.....	15
1.4.6 ADDLY/LEBPRL, 地址 0x1F	16
1.4.7 ADCON3, 地址 0x41A	16
1.4.8 ADCMPH, 地址 0x41B.....	17
2 应用范例.....	17

FT62F08x ADC 应用

1. 12bit ADC 模块

模数转换器（Analog-to-digital Converter, ADC）可将模拟输入信号转换为相应的 12 位二进制表征值。该系列器件采用多个模拟输入复用到一个采样保持电路。采样保持电路的输出与转换器的输入相连接。转换器通过逐次逼近法产生 12 位二进制值，并将转换结果保存在 ADC 结果寄存器（ADRESL:ADRESH）中。ADC 参考电压可用软件选择为 VDD、外部参考电压或内部产生的参考电压。ADC 可在转换完成时产生中断。该中断可用于将器件从休眠唤醒。

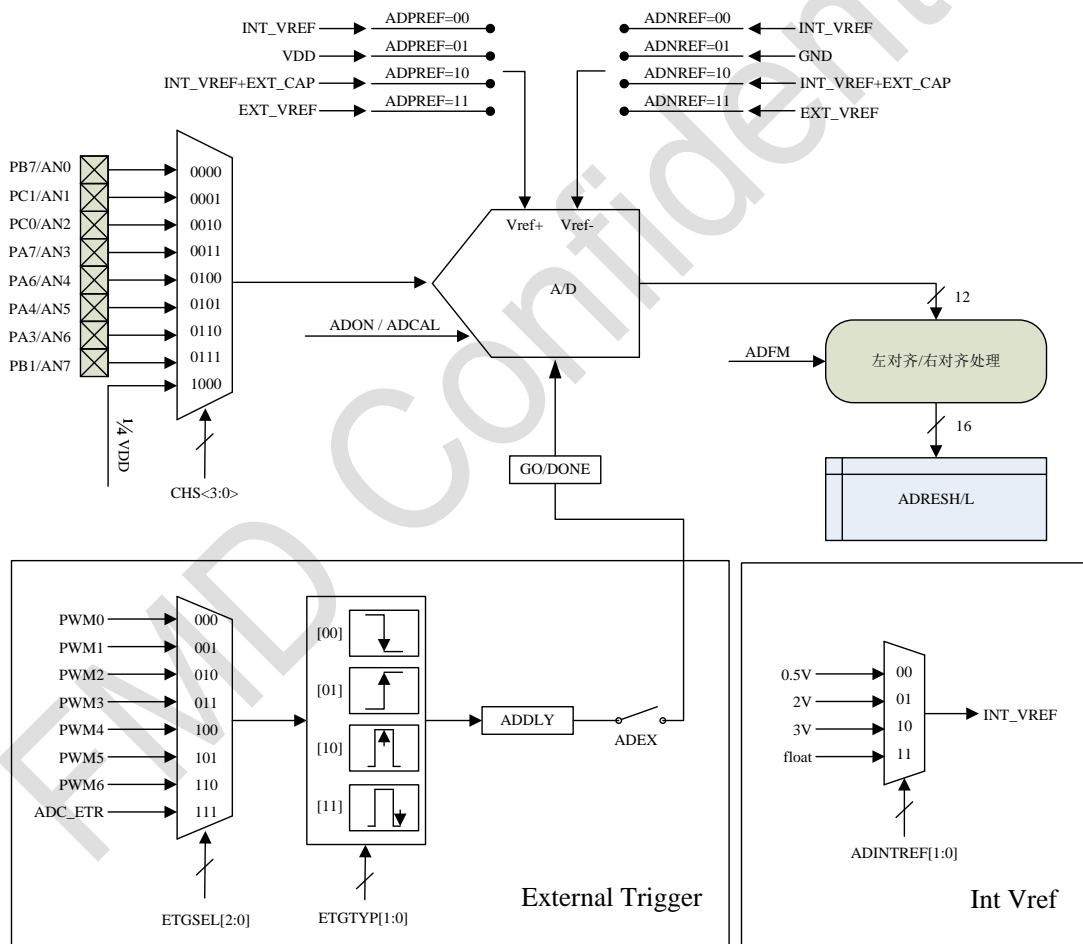


图 9.1 ADC 原理框图

1.1 ADC 的配置

配置和使用 ADC 时，必须考虑以下功能：

- 校准 ADC
- 端口配置
- 通道选择
- 触发方式选择
- 触发源选择
- 触发类型选择
- 触发延时配置
- ADC 参考电压的选择
- ADC 转换时钟源
- 中断控制
- 转换结果的格式
- 阈值比较

注意：在进行各项配置更改的时候，需要确保 AD 转换并未正在进行或外部触发功能未开启。建议在 ADON 关闭时进行更改。

1.1.1 校准 ADC

ADC 在启动转换之前建议至少进行一次校准。校准值会一直保存但不可见，任何复位都会导致其丢失。

ADCON0 寄存器的 ADCAL 设定为 1 可启动自动校准，不可以与 ADON 同时设定为 1。更多信息请参见第 9.2 节“ADC 的工作原理”。

1.1.2 端口配置

ADC 可用于转换模拟和数字信号。转换模拟信号时，应将相关的 TRIS 和 ANSEL 位置 1 将 I/O 引脚应配置为模拟功能。更多信息请参见相应的端口章节。

注意：如果定义为数字输入的引脚上存在模拟电压，会导致输入缓冲器传导过大的电流。

1.1.3 通道选择

ADCON0 寄存器的 CHS 位决定将哪个通道连接到采样保持电路。改变通道时，根据采样稳定的需要可在启动转换前加入一定延时，硬件已固定有 $1.5T_{AD}$ 的采样延时。更多信息请参见第 9.2 节“ADC 的工作原理”。

1.1.4 触发方式选择

ADCON0 寄存器的 ADEX 位决定是否使用外部触发信号。

若 ADEX=0 时，ADGO 可由程序置位，AD 转换完成自动清零。

若 ADEX=1 时，ADGO 将由外部硬件触发置位，AD 转换完成清零。

注意：若选择了前沿消隐触发 ADC，即 LEBADT 设为 1 时，需要先置位 ADEX 和 ADON。

1.1.5 触发源选择

在设定 ADEX 后，ADCON2 寄存器的 ETGSEL 位决定使用哪个外部触发信号。其中可选 I/O 引脚触发，需要配置相关寄存器。具体请参见相应的端口章节。

1.1.6 触发类型选择

ADCON2 寄存器的 ETGTYP 位决定外部触发信号的触发类型。

其中选择 PWM 的中点或终点类型时，触发源将会默认选择 Timer1 中心对齐的 PWM 输出信号。具体请参见相应的 Timer1 章节。

1.1.7 触发延时配置

ADCON2 寄存器的 ADDLY.8 位和 ADDLY 寄存器组成 9 位延时计数器，共同决定外部触发信号的触发延时时间。由于需要同步异步信号，实际延迟时间为： $(ADDLY+6)/F_{ADC}$ 。

注意：若选择了前沿消隐触发功能时，则实际延迟时间为： $(ADDLY+3)/F_{TIM1} + 3/F_{ADC}$ 。

1.1.8 ADC 参考电压

ADCON1 寄存器的 ADPREF 位提供对正参考电压的控制，ADNREF 位提供对负参考电压的控制。正/负参考电压可以是内部参考电压、VDD/GND、内部参考电压加外部电容、外部参考电压。正/负参考电压可以有各种组合，但不可以同时选择内部参考电压。若发生则强制负参考电压选择 GND。

ADCON2 寄存器的 ADINTREF 位提供对内部参考电压的控制。内部参考电压可以选择 0.5V、2V、3V 或者悬空。

1.1.9 转换时钟

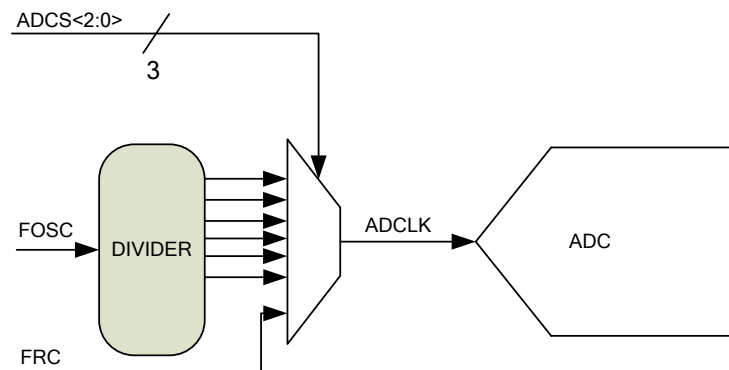


图 9.2 ADC 的时钟配置原理

转换时钟源可通过 ADCON1 寄存器的 ADCS 位用软件选择。有以下 7 种时钟选项：

- $F_{OSC}/2$
- $F_{OSC}/4$
- $F_{OSC}/8$
- $F_{OSC}/16$
- $F_{OSC}/32$
- $F_{OSC}/64$
- F_{RC} （内部慢时钟振荡器）

完成一位（bit）的转换时间定义为 T_{AD} 。完成 12 位转换需要 15 个 T_{AD} 周期（包括 $1.5T_{AD}$ 的采样时间和 $1T_{AD}$ 的数据传输处理时间），如图 9.3 和 9.6 所示。

进行正确的转换必须满足相应的 T_{AD} 规范。更多信息请参见第 21 节“电气特性”中的 A/D 转换要求。表 9.1 所示为正确选择 ADC 时钟的示例。

注意：

1. 除非使用的是 F_{RC} ，否则任何系统时钟频率的变化均会改变 ADC 时钟频率，这将对 ADC 结果产生负面影响；
2. F_{RC} 可以是 256KHz 或者是 32KHz，取决于 LFMOD 为何值；

ADC 时钟周期 (T_{AD})		系统时钟频率 (F_{OSC})			
ADC 时钟源	ADCS<2:0>	16MHz	8MHz	4MHz	1MHz
$F_{OSC}/2$	000	125ns	250ns	500ns	2.0us
$F_{OSC}/4$	100	250ns	500ns	1.0us	4.0us
$F_{OSC}/8$	001	0.5us	1.0us	2.0us	8.0us
$F_{OSC}/16$	101	1.0us	2.0us	4.0us	16.0us
$F_{OSC}/32$	010	2.0us	4.0us	8.0us	32.0us
$F_{OSC}/64$	110	4.0us	8.0us	16.0us	64.0us
F_{RC}	x11	> 72ns	> 72ns	> 72ns	> 72ns

表 9.1 ADC 时钟周期和器件工作频率

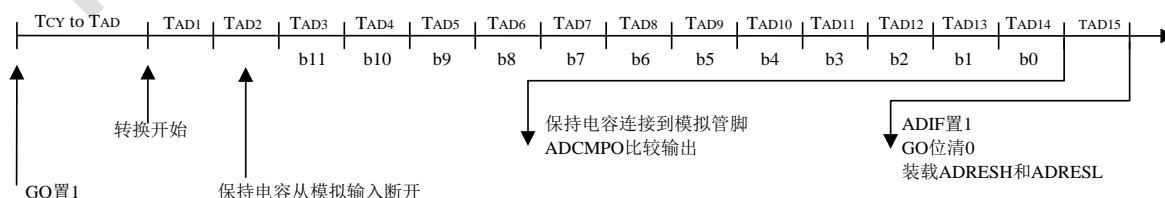


图 9.3 模数转换 T_{AD} 周期

1.1.10 中断

ADC 模块可使中断在模数转换完成时产生。ADC 中断标志为 PIR1 寄存器中的 ADIF 位。ADC 中断使能为 PIE1 寄存器中的 ADIE 位。ADIF 位必须用软件置 1 清零。

- 注意：**
- 1、无论 ADC 中断是否被打开，ADIF 位在每次正常转换完成时均置 1。
 - 2、自动校准完成、软件停止 AD 转换都不会置位 ADIF。
 - 3、仅当在选择了 F_{RC} 振荡器或打开 SYSON bit 时，ADC 才能在休眠期间工作。

器件工作或处于休眠状态时均可产生中断。如果器件处于休眠状态，中断可唤醒器件。从休眠唤醒时，始终执行 SLEEP 指令后的那条指令。如果用户试图唤醒器件并恢复顺序执行代码，必须禁止全局中断。如果允许全局中断，代码执行将转至中断服务程序。

1.1.11 转换结果的格式

12 位 A/D 转换结果有两种格式，即左对齐和右对齐。ADCON1 寄存器的 ADFM 位控制输出格式。

AD 自动校准值也受输出格式影响。

图 9.4 所示为两种输出格式。

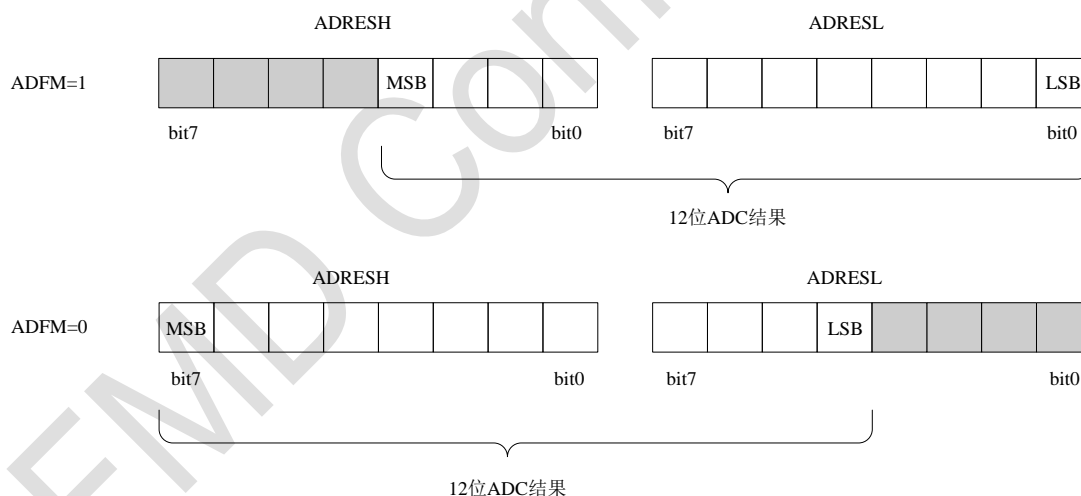


图 9.4 ADC 转换结果格式示意

1.1.12 阈值比较

ADCMPPH 寄存器为 ADC 结果比较阈值，ADCON3 寄存器的 ADCMPEN 位控制比较功能使能，ADCMPOP 位控制比较极性，ADCMPO 指示比较结果。

ADC 可以在每次转换完成时进行比较。比较结果会一直保持，直到下次转换完成被更新。

ADCMPPEN 或 ADON 的清零可以关闭比较功能或 AD 模块，同时可以清零 ADCMPO。进入睡眠不会清零 ADCMPO。

在每次比较完成时可以产生故障刹车事件，由 ADCON3 寄存器的 ADFBEN 控制。

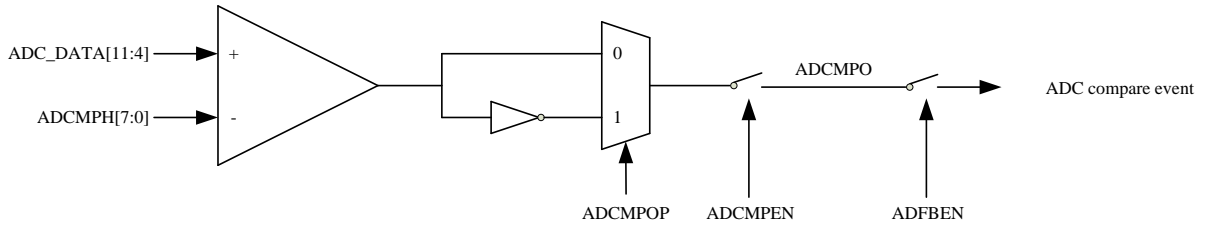


图 9.5 ADC 阈值比较功能框图

1.2 ADC 的工作原理

1.2.1 启动自动校准

要启动 ADC 模块的自动校准功能，必须将 ADCON0 寄存器的 ADCAL 位置 1。校准完成后会自动将 ADCAL 清零，并且将校准值更新到 ADRESH/L。校准完成后 ADC 模块处于已校准状态，直到器件掉电或复位前均有效。

注意： ADCAL 不能跟 ADON 同时置 1。

1.2.2 启动转换

要使能 ADC 模块，必须将 ADCON0 寄存器的 ADON 位置 1。

若 ADEX=0 时，将 ADCON0 寄存器的 GO/DONE 位置 1 将启动 AD 转换。

若 ADEX=1 时，需要外部触发信号才能启动，并且硬件置位 GO/DONE 位，程序置位 ADGO 无效。

注意：

1. 不应在打开 ADC 的那条指令中将 GO/DONE 位置 1。请参见第 9.2.7 节“A/D 转换步骤”
2. 不应在启动 ADC 转换后或等待外部触发时更改 AD 配置。
3. 置位 ADGO 后需要等待一个系统周期才可读回 ADGO 标志。

1.2.3 转换完成

转换完成时，ADC 模块将：

- 将 GO/DONE 位清零
- 将 ADIF 标志位置 1
- 用新的转换结果更新 ADRESH:ADRESL 寄存器
- 若使能阈值比较功能，则更新 ADCMPO 比较结果

1.2.4 终止转换

如果转换必须在完成前被终止，可用软件将 GO/DONE 清零。ADRESH:ADRESL 将使用部分完成的模数转换结果进行更新。未完成位将用最后转换的一位填充。

终止转换需要处理时间，实际处理时间为 $4/F_{AD}$ ，即在这个时间后才会更新 AD 转换结果，若终止时 AD 还未开始转换，则不更新 AD 转换结果。

终止转换不会产生中断。

注意：器件复位将强制所有寄存器回到其复位状态。这样，ADC 模块就被关闭，并且任何待处理的转换均被终止。

1.2.5 休眠模式下 ADC 的工作

ADC 模块可在休眠期间工作，这要求将 ADC 时钟源置于 F_{RC} 选项或打开 SYSON 位。

ADC 需要等待 $4 \cdot T_{AD}$ 后才开始转换。这允许软件在设置 ADGO 后，执行一个 SLEEP 指令置 MCU 于 SLEEP 模式，从而降低 ADC 转换期间的系统噪声。通过配置 ADC 时钟为 F_{RC} 和清零 SYSON，可进一步降低系统噪声。

如果允许 ADC 中断，转换完成后器件将从休眠唤醒。如果禁止 ADC 中断，ADC 模块在转换完成后关闭，尽管 ADON 位保持置 1 状态。

如果 ADC 时钟源不是 F_{RC} 并且 SYSON 未打开，执行一条 SLEEP 指令将使当前转换强制中止，ADC 模块被直接关闭，尽管 ADON 位保持置 1 状态。

如果需要在休眠模式下搭配其它模块一起使用，具体请参见相应的功能配置章节，如 TIMER、GPIO、CLK 管理模块。

1.2.6 外部触发器

除了通过软件启动 AD 转换外，还可以通过硬件触发方式启动 AD 转换。在 ADEX 置 1 后，可选择 PWM 通道的边沿或周期、管脚边沿等触发信号自动触发启动 AD 转换（硬件自动置位 GO/DONE）。这允许在没有软件介入的情况下，定期进行 AD 转换。

通过 ETGSEL (ADCON2[2:0]) 和 ETGTYP (ADCON2[5:4]) 设置来选择触发源和触发类型。同时，还可以在外部触发信号与启动 AD 转换之间插入触发延时。

在 AD 模块转换过程中 (GO/DONE = 1)，任何软件或硬件触发信号都是无效的。若这时停止转换，清零 ADGO 并不会停止触发延时计数。可清零 ADEX 停止触发延时计数。

只有配置 TIMER 为 PWM 输出模式并且使能 PWM 输出时，才会产生 AD 触发信号。更多信息请参见相应的 TIMER 章节。

注意：

当 LEBEN=1 时，外部触发器被禁止。这种情况下，可以选择把 LEBADT 置 1，此时 ADON 和 ADEX 必须设置为 1。在消隐周期结束后会触发一次 AD 转换（硬件自动置位 GO/DONE）。

1.2.7 A/D 转换步骤

以下是使用 ADC 进行模数转换的步骤示例：

1. 配置端口：
 - 禁止引脚输出驱动器（见 TRIS 寄存器）
 - 将引脚配置为模拟
2. 配置 ADC 模块：
 - 选择 ADC 转换时钟
 - 配置参考电压
 - 选择 ADC 输入通道
 - 配置触发源、类型及延时
 - 选择转换结果的格式
 - 配置 ADC 结果阈值比较
 - 打开 ADC 模块
3. 配置 ADC 中断（可选）：
 - 将 ADC 中断标志清零
 - 允许 ADC 中断
 - 允许外设中断
 - 允许全局中断
4. 等待所需稳定时间 $T_{ST}^{(1)}$ ；
5. 将 GO/DONE 置 1 启动转换或等待硬件触发；
6. 等待一个系统周期才可回读 GO/DONE；
7. 通过以下情况之一等待 ADC 转换完成：
 - 查询 GO/DONE 位
 - 等待 ADC 中断（允许中断时）
8. 读取 ADC 结果；
9. 将 ADC 中断标志清零（在允许了中断的情况下这一步是必需的）。

以下是一段示例代码：

```

BANKSEL  ADCON1          ;
MOVLW    B'11110101'      ;Right justify, ADC Frc clock
MOVWF    ADCON1          ;Vref+ VDD , Vref- GND
BANKSEL  TRISB            ;
BSR      TRISB,7          ;Set PB7 to input
BANKSEL  ANSELA           ;
BSR      ANSELA,0         ;Set PB7 to analog
BANKSEL  ADCON0           ;
MOVLW    B'00000001'      ;Select channel AN0,
MOVWF    ADCON0          ;Turn ADC On
BSR      ADCON0,ADGO       ;Start conversion (ADC stable time)
NOP      ;ADGO ReadBack WaitTime
BTSC     ADCON0,ADGO       ;Is conversion done?
GOTO     $-1              ;No, test again
BANKSEL  ADRESH           ;

```

```

MOVF      ADRESH,W           ;Read upper 4 bits
MOVWF     RESULTHI           ;store in GPR space
BANKSEL   ADRESL             ;
MOVF      ADRESL,W           ;Read lower 8 bits
MOVWF     RESULTLO           ;Store in GPR space

```

注意:

1. T_{ST} 时间是 ADC 的稳定时间(15 个 T_{AD})，每次关闭 AD 转换再启动时需要等待 T_{ST} ；

1.3 A/D 采集时间要求

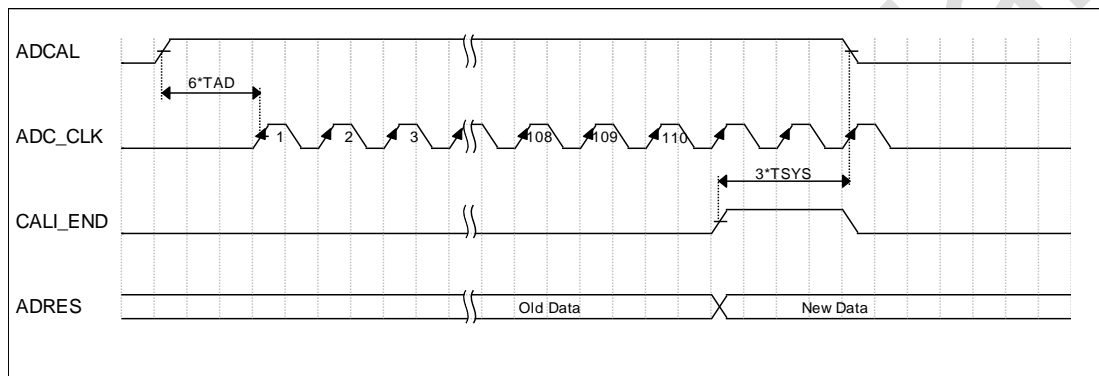


图 9.6 ADC 自校准时序图

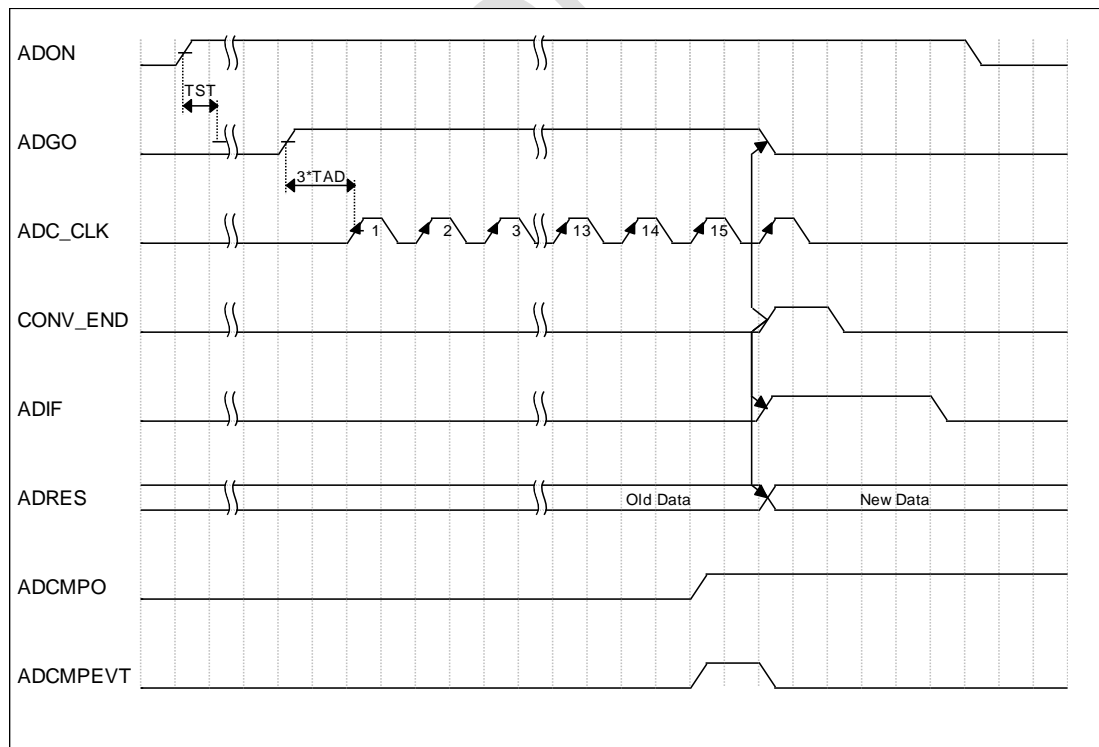


图 9.7 ADC 软件触发转换时序图

为了使 ADC 达到规定的精度，必须使充电保持电容（CHOLD）充满至输入通道的电平。模拟

输入模型请参见图 9.8。源阻抗（RS）和内部采样开关（RSS）阻抗直接影响电容 CHOLD 的充电时间。采样开关（RSS）阻抗随器件电压（VDD）的变化而变化，参见图 12.6。建议模拟信号源的最大阻抗为 10kΩ。采集时间随着源阻抗的降低而缩短。在选择（或改变）模拟输入通道后，必须在开始转换前完成采集。

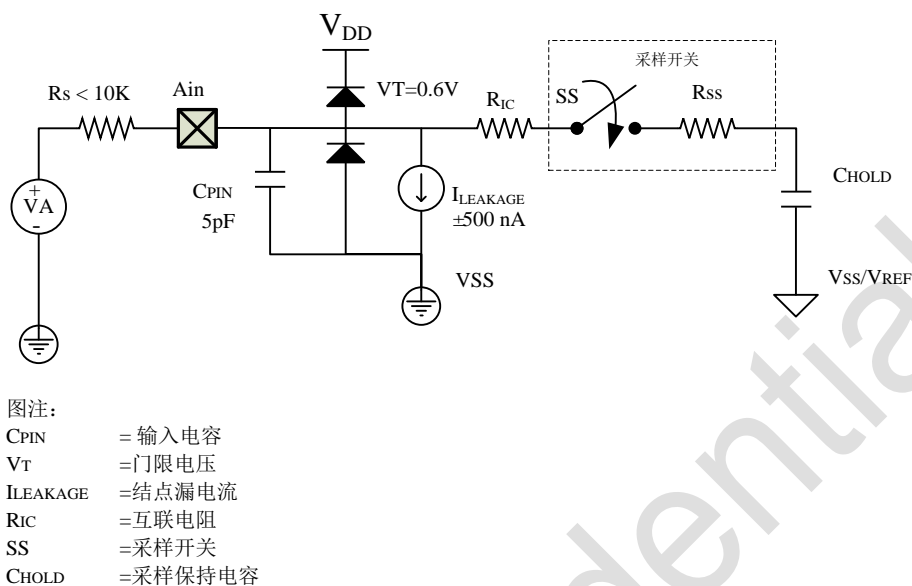


图 9.8 模拟输入模型

1.4 与 ADC 相关寄存器汇总

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位 值
ADRESL	0x09B	A/D 结果寄存器的低字节								0000 0000
ADRESH	0x09C	A/D 结果寄存器的高字节								0000 0000
ADCON0	0x09D	CHS<3:0>				ADCAL	ADEX	GO/DONE	ADON	0000 0000
ADCON1	0x09E	ADFM	ADCS<2:0>			ADNREF<1:0>		ADPREF<1:0>		0000 0000
ADCON2	0x09F	ADINTREF<1:0>		ETGTYP<1:0>		ADDLY.8	ETGSEL<2:0>			0000 0000
ADDLY	0x01F	ADDLY<7:0> / LEBPRL<7:0>								0000 0000
ADCON3	0x41A	ADFBEN	ADCMPOP	ADCMPEN	ADCMPO	LEBADT	—	ELVDS<1:0>		0 0 0 0 0 - 0 0
ADCMPH	0x41B	ADCMPH<7:0>								0000 0000
LEBCON	0x41C	LEBEN	LEBCH		EDGS		BKS2	BKS1	BKS0	0000

								0000
--	--	--	--	--	--	--	--	------

注：ADCON2、ADDLY、ADCON3、LEBCON 即使在 PCKEN 的 ADCEN 位为 0 时也可以读写。

1.4.1 ADRESL, 地址 0x9B

Bit	7	6	5	4	3	2	1	0
Name	ADRESL<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADRESL<7:0>	ADC 结果寄存器的低字节 ADFM=0 时, ADRESL[7:4]为 12 位转换结果的低 4 位, 其余为 0。 ADFM=1 时, ADRESL[7:0]为 12 位转换结果的低 8 位。

1.4.2 ADRESH, 地址 0x9C

Bit	7	6	5	4	3	2	1	0
Name	ADRESH<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADRESH<7:0>	ADC 结果寄存器的高字节 ADFM=0 时, ADRESH[7:0]为 12 位转换结果的高 8 位。 ADFM=1 时, ADRESH[3:0]为 12 位转换结果的高 4 位, 其余为 0。

1.4.3 ADCON0, 地址 0x9D

Bit	7	6	5	4	3	2	1	0
Name	CHS<3:0>				ADCAL	ADEX	GO/DONE	ADON
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:4	CHS<3:0>	模拟通道选择位 0000 = AN0

		0001 = AN1 0010 = AN2 0011 = AN3 0100 = AN4 0101 = AN5 0110 = AN6 0111 = AN7 1000 = 1/4 V _{DD} 其余保留
3	ADCAL	ADC 自动校准使能位（ADON 为 0 时可设定） 该位由软件设置来启动 ADC 校准。当校准完成后，由硬件清零。 0 = 校准完成。 1 = 写 1 时校准 ADC，读为 1 时意味着校准仍在进行中。
2	ADEX	ADC 触发信号类型选择 该位决定启动 ADC 的触发条件 0 = 当软件设定 GO/DONE 位，启动 AD 转换 1 = 需要外部触发信号触发才可启动 AD 转换，触发事件置位 GO/DONE 位。 外部触发信号条件由寄存器 ETGSEL<2:0>和 ETGTYP<1:0>决定。
1	GO/DONE	AD 转换状态位（硬件触发事件直接置位） 将该位置 1 可启动 A/D 转换周期。当 A/D 转换完成以后，该位由硬件自动清零。 0 = A/D 转换完成/未进行。 1 = A/D 转换正在进行或硬件触发延时正在计数。
0	ADON	ADC 使能位 0 = ADC 被禁止且不消耗工作电流 1 = ADC 被使能

1.4.4 ADCON1, 地址 0x9E

Bit	7	6	5	4	3	2	1	0
Name	ADFM	ADCS<2:0>			ADNREF<1:0>		ADPREF<1:0>	
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	ADFM	ADC 结果格式选择位 1 = 右对齐。装入转换结果时，ADRESH 的高 4 位被设置为 0。 0 = 左对齐。装入转换结果时，ADRESL 的低 4 位被设置为 0。
6:4	ADCS<2:0>	ADC 转换时钟选择位 000 = F _{OSC} /2 001 = F _{OSC} /8 010 = F _{OSC} /32 011 = F _{RC} （由专用 RC 振荡器提供时钟）

		$100 = F_{OSC}/4$ $101 = F_{OSC}/16$ $110 = F_{OSC}/64$ $111 = F_{RC}$ (由专用 RC 振荡器提供时钟)
3:2	ADNREF	ADC 负参考电压配置位 (使用 PB6 连接外部参考电压或外部电容) $00 = \text{Int Vref}$ (内部参考电压) $01 = \text{GND}$ $10 = \text{Int Vref} + \text{Ext Cap}$ (内部参考电压 + 外部电容) $11 = \text{Ext Vref}$ (外部参考电压)
1:0	ADPREF	ADC 正参考电压配置位 (使用 PB5 连接外部参考电压或外部电容) $00 = \text{Int Vref}$ (内部参考电压) $01 = V_{DD}$ $10 = \text{Int Vref} + \text{Ext Cap}$ (内部参考电压 + 外部电容) $11 = \text{Ext Vref}$ (外部参考电压)

1.4.5 ADCON2, 地址 0x9F

Bit	7	6	5	4	3	2	1	0
Name	ADINTREF<1:0>		ETGTYP<1:0>		ADDLY.8	ETGSEL<2:0>		
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:6	ADINTREF<1:0>	ADC 内部参考电压配置位 $00 = 0.5V$ $01 = 2V$ $10 = 3V$ $11 = \text{float}$ (悬空)
5:4	ETGTYP<1:0>	外部触发信号类型选择 当 ADEX 置 1, 该位决定响应外部触发的类型 $00 = \text{PWM}$ 或 ADC_ETR 脚的下降沿 $01 = \text{PWM}$ 或 ADC_ETR 脚的上升沿 $10 =$ 一个 PWM 周期的中点 $11 =$ 一个 PWM 周期的终点 注: PWM 周期中点或终点触发仅适用于中心对齐模式的 PWM 输出
7	ADDLY.8 /LEBPR9	ADC 外部触发延时计数器阈值 第 8 位 详见 ADDLY 寄存器描述
2:0	ETGSEL<2:0>	外部触发源选择 当 ADEX 为 1, 该位选择外部触发 ADC 的来源 选择 PWM 源时需要配置 TIMER 为 PWM 输出模式并使能输出。 $000 = \text{PWM0}$ $001 = \text{PWM1}$

		010 = PWM2 011 = PWM3 100 = PWM4 101 = PWM5 110 = PWM6 111 = ADC_ETR
--	--	---

1.4.6 ADDLY/LEBPRL, 地址 0x1F

Bit	7	6	5	4	3	2	1	0
Name	ADDLY<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADDLY<7:0>	<p>ADC 外部触发启动延时计数器阈值低位</p> <p>该 8 位寄存器与 ADCON2.7 组成 9 位计数器, 用于在外部触发启动 ADC 之前加入一段延迟。延迟计数器结束再开始 ADC 转换</p> <p>外部延迟时间 = (ADDLY+6)/F_{ADC}</p> <p>注, 该延时仅当 ADEX 置 1 时有效。如果启用 PWM 输出触发 ADC 功能, 在 PWM 运行过程中不得更改 ADDLY 计数值。同时复用为前沿消隐计数阈值</p>

1.4.7 ADCON3, 地址 0x41A

Bit	7	6	5	4	3	2	1	0
Name	ADFBEN	ADCMPOP	ADCMPEM	ADCMPO	LEBADT	—	ELVDS	
Reset	0	0	0	0	0	—	0	0
Type	RW	RW	RW	RO	RW	RO-0	RW	RW

Bit	Name	Function
7	ADFBEN	<p>ADC 比较结果响应故障刹车使能</p> <p>0 = 禁止</p> <p>1 = ADC 触发故障刹车功能使能</p>
6	ADCMPOP	<p>ADC 比较器输出极性选择位</p> <p>0 = 若 ADC 结果的高八位大于或等于 ADCMPH[7:0], ADCMPO 为 1</p> <p>1 = 若 ADC 结果的高八位小于 ADCMPH[7:0], ADCMPO 为 1</p>
5	ADCMPEM	<p>ADC 结果比较使能位</p> <p>0 = ADC 结果比较功能关闭</p> <p>1 = ADC 结果比较功能打开</p>
4	ADCMPO	ADC 比较结果输出位

		该位输出 ADCMPPOP 设定的比较输出结果。每次 AD 转换结束都会更新输出
3	LEBADT	前沿消隐周期结束后，ADC 触发使能 1 = 触发 ADC 转换 0 = 不触发 ADC 转换
2	N/A	保留：读为 0
1:0	ELVDS	外部 LVD 管脚输入选择，只有当 LVDM 为 1 时才有效 00 = ELVD0 01 = ELVD1 10 = ELVD2 11 = ELVD3

1.4.8 ADCMPH，地址 0x41B

Bit	7	6	5	4	3	2	1	0
Name	ADCMPH<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADCMPH<7:0>	ADC 比较阈值 仅 8 位，用于 ADC 结果高 8 位比较。

2 应用范例

```

/*****
/* 文件名: TEST_62F08x_ADC.c
* 功能: FT62F08x-ADC 功能演示
* IC: FT62F088
* 内部: 16M
* empro: 500
* 说明: 程序采样 AN0 口变阻器的 AD 值并计算其电压
*
*
* 参考原理图 TEST_62F08x_sch.pdf
*/
/*****

#include "SYSCFG.h"

/*****
//宏定义 *****/

```

```
#define uchar      unsigned char
#define uint       unsigned int
#define ulong      unsigned long
```

```
volatile uint  adcData;
volatile uint  theVoltage;
```

```
/*-----
```

```
* 函数名:  interrupt ISR
* 功能:   中断处理，包括定时器 0 中断和外部中断
* 输入:   无
* 输出:   无
```

```
-----*/
```

```
void interrupt ISR(void)          //PIC_HI-TECH 使用
```

```
{
```

```
}
```

```
/*-----
```

```
* 函数名:  POWER_INITIAL
* 功能:   上电系统初始化
* 输入:   无
* 输出:   无
```

```
-----*/
```

```
void POWER_INITIAL (void)
```

```
{
```

```
    OSCCON = 0B01110001; //16MHZ 1:1
```

```
    //BIT7~BIT4: 主 时 钟 （ 系 统 时 钟 ） 分 频 比 选 择 。
```

```
0111(1:1),0110(1:2),0101(1:4),0100(1:8),0011(1:16),0010,(1:32),0001(1:64),1xxx(1:128),0000(32kHz LIRC)
```

```
    //BIT3:振荡器起振超时状态位。1: 器件运行在 FOSC<2:0>指定的外部时钟之下；0: 器件运行在内部振荡器之下
```

```
    //BIT2:高速内部时钟状态。 1: HIRC is ready; 0: HIRC is not ready
```

```
    //Bit1: 低速内部时钟状态。1: LIRC is ready; 0: LIRC is not ready
```

```
    //Bit0:系统时钟选择位。1: 系统时钟选择为内部振荡器；0: 时钟源由 FOSC<2:0>决定
```

```
    PCKEN |=0B00000001;      //AD 模块时钟使能
```

```
    INTCON = 0;              //暂禁止所有中断
```

```
    PORTA = 0B00000000;
```

```
    TRISA = 0B11111111;      //PA 输入输出 0-输出 1-输入
```

```
    PORTB = 0B00000000;
```

```
    TRISB = 0B11111111;      //PB 输入输出 0-输出 1-输入
```

```
    PORTC = 0B00000000;
```

```

TRISC = 0B11111111;    //PC 输入输出 0-输出 1-输入
PORTD = 0B00000000;
TRISD = 0B11111111;    //PD 输入输出 0-输出 1-输入

WPUA = 0B00000000;    //PA 端口上拉控制 1-开上拉 0-关上拉
WPUB = 0B00000000;    //PB 端口上拉控制 1-开上拉 0-关上拉
WPUC = 0B00000000;    //PC 端口上拉控制 1-开上拉 0-关上拉
WPUD = 0B00000000;    //PD 端口上拉控制 1-开上拉 0-关上拉

WPDA = 0B00000000;    //PA 端口上拉控制 1-开下拉 0-关下拉
WPDB = 0B00000000;    //PB 端口上拉控制 1-开下拉 0-关下拉
WPDC = 0B00000000;    //PC 端口上拉控制 1-开下拉 0-关下拉
WPDD = 0B00000000;    //PD 端口上拉控制 1-开下拉 0-关下拉

PSRC0 = 0B11111111;    //PORTA,PORTB 源电流设置最大
//BIT7~BIT6:PORTB[7:4]源电流能力控制,BIT5~BIT4:PORTB[3:0]源电流能力控制
//BIT3~BIT2:PORTA[7:4]源电流能力控制,BIT1~BIT0:PORTA[3:0]源电流能力控制

PSRC1 = 0B11111111;    //PORTC,PORTD 源电流设置最大
//BIT7~BIT6:PORTD[7:4]源电流能力控制,BIT5~BIT4:PORTD[3:0]源电流能力控制
//BIT3~BIT2:PORTC[7:4]源电流能力控制,BIT1~BIT0:PORTC[3:0]源电流能力控制

PSINK0 = 0B11111111;    //PORTA 灌电流设置最大 0:最小, 1:最大
PSINK1 = 0B11111111;    //PORTB 灌电流设置最大 0:最小, 1:最大
PSINK2 = 0B11111111;    //PORTC 灌电流设置最大 0:最小, 1:最大
PSINK3 = 0B11111111;    //PORTD 灌电流设置最大 0:最小, 1:最大
}
/*-----
* 函数名称: DelayUs
* 功能: 短延时函数 --16M-2T--大概快 1%左右.
* 输入参数: Time 延时时间长度 延时时长 Time Us
* 返回参数: 无
-----*/
void DelayUs(unsigned char Time)
{
    unsigned char a;
    for(a=0;a<Time;a++)
    {
        NOP();
    }
}
/*-----
* 函数名: ADC_INITIAL
* 功能: ADC 初始化

```

* 输入： 无

* 输出： 无

-----*/

void ADC_INITIAL(void)

{

ANSELA = 0B00000001; //模拟口设置, AN0 为模拟管脚

ADCON1 = 0B11100100; //右对齐, 转换时钟 Fosc/64, 负参考电压 GND, 正参考电压
内部电压(2V)

//BIT7:

//1 = 右对齐。装入转换结果时, ADRESH 的高 4 位被设置为 0;

//0 = 左对齐。装入转换结果时, ADRESL 的低 4 位被设置为 0。

//BIT6~BIT4:ADC 转换时钟选择位

//000 = FOSC/2

//001 = FOSC/8

//010 = FOSC/32

//011 = FRC (由专用 RC 振荡器提供时钟)

//100 = FOSC/4

//101 = FOSC/16

//110 = FOSC/64

//111 = FRC (由专用 RC 振荡器提供时钟)

//BIT3~BIT2:ADC 负参考电压配置位 (使用 PB6 连接外部参考电压或外部电容)

//00 = Int Vref (内部参考电压)

//01 = GND

//10 = Int Vref + Ext Cap (内部参考电压 + 外部电容)

//11 = Ext Vref (外部参考电压)

//BIT1~BIT0 ADC 正参考电压配置位 (使用 PB5 连接外部参考电压或外部电容)

//00 = Int Vref (内部参考电压)

//01 = VDD

//10 = Int Vref + Ext Cap (内部参考电压 + 外部电容)

//11 = Ext Vref (外部参考电压)

ADCON0 = 0B00000000; //选择 AD 转换通道 0, 使能 ADC

//BIT7~BIT4:ADC 模拟通道选择位

//0000 = AN0

//0001 = AN1

//0010 = AN2

//0011 = AN3

//0100 = AN4

//0101 = AN5

//0110 = AN6
//0111 = AN7
//1000 = 1/4 VDD
//其余保留

//BIT3:该位由软件设置来启动 ADC 校准。当校准完成后，由硬件清零。

//0 = 校准完成。

//1 = 写 1 时校准 ADC，读为 1 时意味着校准仍在进行中。

//BIT2: ADC 触发信号类型选择

//该位决定启动 ADC 的触发条件

//0 = 当软件设定 GO/DONE 位，启动 AD 转换

//1 = 需要外部触发信号触发才可启动 AD 转换，触发事件置位 GO/DONE 位。

//外部触发信号条件由寄存器 ETGSEL<2:0>和 ETGTYP<1:0>决定。

//BIT1:

//0 = A/D 转换完成/未进行。

//1 = A/D 转换正在进行或硬件触发延时正在计数。

//BIT0:使能 ADC

//0 = ADC 被禁止且不消耗工作电流

//1 = ADC 被使能

ADCON2 = 0B01000000; //选择内部正参考电压 2V，无外部触发源

//BIT7~BIT6:ADC 内部参考电压配置位

//00 = 0.5V

//01 = 2V

//10 = 3V

//11 = float (悬空)

//BIT5~BIT4:外部触发信号类型选择

//当 ADEX 置 1，该位决定响应外部触发的类型

//00 = PWM 或 ADC_ETR 脚的下降沿

//01 = PWM 或 ADC_ETR 脚的上升沿

//10 = 一个 PWM 周期的中点

//11 = 一个 PWM 周期的终点

//BIT3:ADC 外部触发延时计数器阈值 第 8 位

//BIT2~BIT0:外部触发源选择

//当 ADEX 为 1，该位选择外部触发 ADC 的来源

//选择 PWM 源时需要配置 TIMER 为 PWM 输出模式并使能输出。

//000 = PWM0

```
//001 = PWM1  
//010 = PWM2  
//011 = PWM3  
//100 = PWM4  
//101 = PWM5  
//110 = PWM6  
//111 = ADC_ETR
```

ADCON3 = 0B00000000;

//BIT7:ADC 比较结果响应故障刹车使能

//0 = 禁止

//1 = ADC 触发故障刹车功能使能

//BIT6:ADC 比较器输出极性选择位

//0 = 若 ADC 结果的高八位大于或等于 ADCMPH[7:0], ADCMPO 为 1

//1 = 若 ADC 结果的高八位小于 ADCMPH[7:0], ADCMPO 为 1

//BIT5:ADC 结果比较使能位

//0 = ADC 结果比较功能关闭

//1 = ADC 结果比较功能打开

//BIT4:ADC 比较结果输出位

//该位输出 ADCMPOP 设定的比较输出结果。每次 AD 转换结束都会更新输出

//BIT3:前沿消隐周期结束后, ADC 触发使能

//1 = 触发 ADC 转换

//0 = 不触发 ADC 转换

//BIT2:保留位

//BIT1~BIT0 外部 LVD 管脚输入选择, 只有当 LVDM 为 1 时才有效

//00 = ELVD0

//01 = ELVD1

//10 = ELVD2

//11 = ELVD3

ADDLY = 0B00000000; //外部触发延时, 没用到

//ADC 外部触发启动延时计数器阈值低位

//该 8 位寄存器与 ADCON2.7 组成 9 位计数器, 用于在外部触发启动 ADC 之前加入一段延迟。延迟计数器结束再开始 ADC 转换

//外部延迟时间 = (ADDLY+6)/FADC

ADCMPH = 0B00000000; //ADC 比较阈值, 仅 8 位, 用于 ADC 结果高 8 位比较。

```

    ADCAL=1;                //校准 ADC, 注意点: 校准要放在配置之后
    NOP();
    while(ADCAL);

    ADON=1; //全能 ADC
}
/*-----
* 函数名: GET_ADC_DATA
* 功能:   读取通道 ADC 值
* 输入:   adcChannel 通道序号
* 输出:   INT 类型 AD 值(单次采样无滤波)
-----*/
uint GET_ADC_DATA (uchar adcChannel)
{
    ADCON0 &= 0B00001111;
    ADCON0 |= adcChannel<<4;           //重新加载通道值
    DelayUs(40);                       //延时等待电压稳定 Tst>10us
    GO = 1;                            //启动 ADC
    NOP();
    NOP();
    while(GO);                         //等待 ADC 转换完成

    return (uint)(ADRESH<<8|ADRESL);  //整合 12 位 AD 值
}
/*-----
* 函数名: main
* 功能:   主函数
* 输入:   无
* 输出:   无
-----*/
void main(void)
{
    POWER_INITIAL();                 //初始化
    ADC_INITIAL();                   //ADC 初始化

    while(1)
    {
        adcData = GET_ADC_DATA(0);   //通道 0 AD 值
        theVoltage = (ulong)adcData*2*1000/4096; //电压放大 1000 倍
        NOP();
        NOP();
    }
}

```

Fremont Micro Devices (SZ) Limited

#5-8, 10/F, Changhong Building, Ke-Ji Nan 12 Road, Nanshan District, Shenzhen, Guangdong 518057

Tel: (86 755) 86117811

Fax: (86 755) 86117810

Fremont Micro Devices (Hong Kong) Limited

#16, 16/F, Blk B, Veristrong Industrial Centre, 34-36 Au Pui Wan Street, Fotan, Shatin, Hong Kong

Tel: (852) 27811186

Fax: (852) 27811144

Fremont Micro Devices (USA), Inc.

42982 Osgood Road Fremont, CA 94539

Tel: (1-510) 668-1321

Fax: (1-510) 226-9918

Web Site: <http://www.fremontmicro.com/>

* Information furnished is believed to be accurate and reliable. However, Fremont Micro Devices, Incorporated (BVI) assumes no responsibility for the consequences of use of such information or for any infringement of patents of other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent rights of Fremont Micro Devices, Incorporated (BVI). Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. Fremont Micro Devices, Incorporated (BVI) products are not authorized for use as critical components in life support devices or systems without express written approval of Fremont Micro Devices, Incorporated (BVI). The FMD logo is a registered trademark of Fremont Micro Devices, Incorporated (BVI). All other names are the property of their respective own.