FT61F13X

ADC_VDD APPLICATION NOTE

目录

FT61F13x ADC_VDD 应用

1 12-bit 模/数转换器 (ANALOG TO DIGITAL CONVERTER, ADC)

ADC 模块可将模拟输入信号转换成 12-bit 的数字信号。ADC 可在不同的时钟速度下运行，并且在高达 850 kHz 的时钟速度(即 48 kHz 的采样率，21 μs/采样) 下仍具有真正的 12-bit 精度。

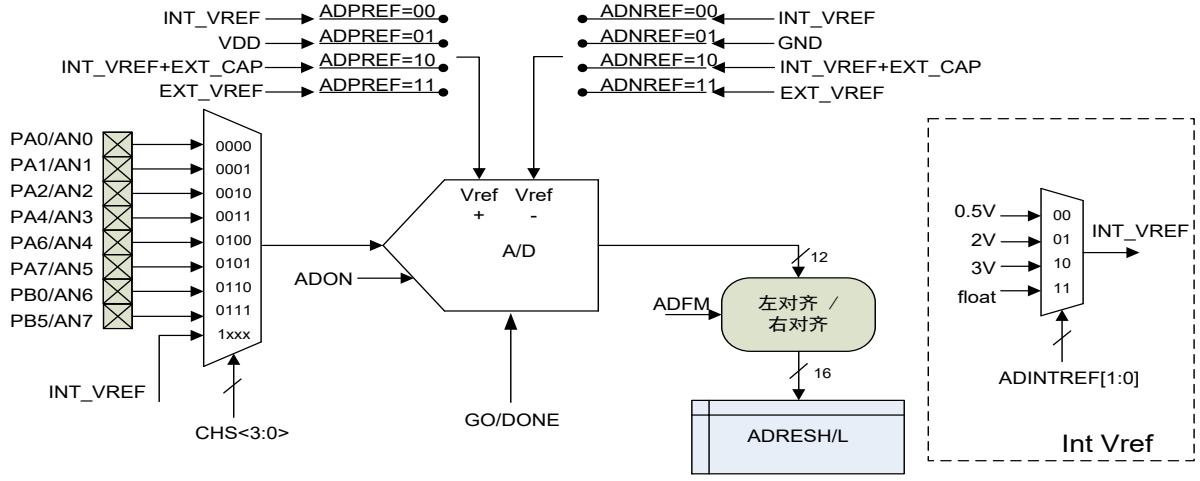

图 1-1 ADC 结构框图

模拟输入信号可选择为 8 个 I/O (ANx) 通道之一或 3 个内部参考电压(Internal $V_{ADC-REF}$)之一。 ADC 由指令、I/O(PA4)或 PWM 触发。在触发和 ADC 采样之间可增加延时或前沿消隐(Leading Edge Blanking, LEB)。

当 ADC 转换完成和/或 ADC 阈值比较结果匹配时，将置位相应的中断标志位，并可触发中断和/或从睡眠中唤醒。

ADC 参考电压($V_{ADC-REF}$)可通过指令选择为 V_{DD}，3 个内部参考电压(0.5V, 2V, 3V)之一，或通过 I/O 输入外部参考电压。

ADC 不需要校准。另外，ADC 转换过程在后台运行，转换期间 CPU 可执行其他指令。

如果 ADC 需要在 SLEEP 下保持运行，且其转换时钟源为 Sysclk 或其分频时，则需通过另外使能所选时钟源为 Sysclk 的 Timers，来使系统时钟 Sysclk 在 SLEEP 下保持运行。当 ADC 的时钟源为 LIRC 时，进入 SLEEP 后 LIRC 将自动开启。

当 ADC 配置为硬件触发(PA4 或 PWM)时，GO/DONE 由硬件触发事件直接置位并启动 A/D 转换，软件置位 GO/DONE 将被忽略。

在高采样率的应用中，使用 ADC 时需注意 3 个时间点：

1. 所选通道开始采样的时刻。
2. 结束采样的时刻。采样保持电路断开前的瞬间，所选通道上的电压值被用于测量转换。
3. 数据转换完成时间。

1.1. ADC 相关寄存器汇总

名称	状态		寄存器	地址	复位值
GIE	全局中断	1 = 使能 (PEIE, ADCIE, ACMPIE 适用) 0 = 全局关闭 (唤醒不受影响)	INTCON[7]	0x0B 0x8B 0x10B 0x18B	RW-0
PEIE	外设总中断	1 = 使能 (ADCIE, ACMPIE 适用) 0 = 关闭 (无唤醒)	INTCON[6]		RW-0
ADCIE	ADC 转换完成中断	1 = 使能 0 = 关闭 (无唤醒)	PIE1[0]	0x8C	RW-0
ADCIF	ADC 转换完成中断标志位	1 = Yes (锁存) 0 = No	PIR1[0]	0x0C	RW-0
ACMPIE	ADC 阈值比较匹配中断	1 = 使能 0 = 关闭 (无唤醒)	PIE1[4]	0x8C	RW-0
ACMPIF	ADC 阈值比较匹配中断标志位	1 = Yes (锁存) 0 = No	PIR1[4]	0x0C	RW-0

表 1-1 ADC 中断使能和状态位

名称	状态	寄存器	地址	复位值
ADRESL	ADC 转换结果低有效位 (LSB) ADFM=0: ADRESL[7:4] = 低 4 位 (其余为"0") ADFM=1; ADRESL[7:0] = 低 8 位	ADRESL[7:0]	0x111	RW-xxxx xxxx
ADRESH	ADC 转换结果高有效位 (MSB) ADFM=0: ADRESH[7:0] = 高 8 位 ADFM=1: ADRESH[3:0] = 高 4 位 (其余为"0")	ADRESH[7:0]	0x112	RW-xxxx xxxx
ADCMPH	ADC 比较阈值 (仅高 8 位，0.4% steps)	ADCMPH[7:0]	0x187	RW-0000 0000
ADDLY / LEBPRL	ADC 延迟/ LEB (非软件触发，ADEX = 1) (此为低 8 位, ADDLY.8 为高有效位) 延迟时间 = (ADDLY+6)/F_{ADC} (如果启用 PWM 输出触发 ADC，在 PWM 运行过程中不得更改 ADDLY)	ADDLY[7:0]	0x188	RW-0000 0000
LEBEN	ADC 触发和 BKIN 的 LEB 使能位 1 = 使能 (当 GO/DONE=1 时进行切换将产生不可预知的结果) 0 = 关闭	LEBCON[7]	0x185	RW-0
LEBCH	LEB 信号源 00 = P1A0 10 = P1C 01 = P1B 11 = P1D	LEBCON[6:5]		RW-00
EDGS	LEB 触发沿 0 = 上升沿 1 = 下降沿	LEBCON[3]		RW-0

名称	状态	寄存器	地址	复位值
CHS	ADC 模拟输入通道 0000 = AN0　　0101 = AN5 0001 = AN1　　0110 = AN6 0010 = AN2　　0111 = AN7 0011 = AN3　　1xxx = (内部 $V_{ADC-REF}$) 0100 = AN4	ADCON0[6:3]	0x113	RW-0000
ADEX	ADC 触发条件 (GO/DONE) 1 = 由 PA4 或 PWM 置位 GO/DONE (硬件触发) 0 = 由指令置位 GO/DONE (软件触发)	ADCON0[2]		RW-0
GO/DONE	ADC 转换启动和状态位 1 = 由软件, PA4 或 PWM 启动 A/D 转换 (转换完成后自动清零) 0 = 转换完成 / 未进行转换	ADCON0[1]		RW-0
ADON	1 = ADC 使能 0 = ADC 关闭 (无电流消耗)	ADCON0[0]		RW-0
ADFM	A/D 转换结果格式 (参阅 “ADRESH”) 1 = 右对齐 0 = 左对齐	ADCON1[7]	0x114	RW-0
ADCS	ADC 转换时钟源 TSEL = 2T　　TSEL = 4T 000 = SysClk/2　　000 = SysClk/4 001 = SysClk/8　　001 = SysClk/16 010 = SysClk/32　　010 = SysClk/64 011 = SysClk　　011 = SysClk/2 100 = SysClk/4　　100 = SysClk/8 101 = SysClk/16　　101 = SysClk/32 110 = SysClk/64　　110 = SysClk/128 111 = LIRC　　111 = LIRC	ADCON1[6:4]		RW-000
ADNREF	$V_{ADC-REF}$ – (负参考电压) 00 = 内部 $V_{ADC-REF}$ 01 = GND 10 = 内部 $V_{ADC-REF}$ + 外部电容 Cap 11 = 外部参考电压 (I/O)	ADCON1[3:2]		RW-00
ADPREF	$V_{ADC-REF}$ + (正参考电压) 00 = 内部 $V_{ADC-REF}$ 01 = V_{DD} 10 = 内部 $V_{ADC-REF}$ + 外部电容 Cap 11 = 外部参考电压 (I/O)	ADCON1[1:0]		RW-00
ADINTREF	内部 $V_{ADC-REF}$ 00 = 0.5　　10 = 3.0 01 = 2.0　　11 = (未连接)	ADCON2[7:6]	0x115	RW-00

名称	状态	寄存器	地址	复位值
ETGTYP	外部触发沿 (当 ADEX=1 时适用) 00 = (PWM 或 PA4-ADC_ETR) 下降沿 01 = (PWM 或 PA4-ADC_ETR) 上升沿	ADCON2[5:4]		RW-00
ADDLY.8 / LEBPR9	ADC 延迟计数器或 LEB 计数器的第 8 位 (参阅 “ADDLY”)	ADCON2[3]		RW-0
ETGSEL	外部触发源 (当 ADEX=1 时适用) 000 = P1A0　100 = P1D 001 = P1A0N　101 = ADC_ETR 010 = P1B　110 = (无) 011 = P1C　111 = (无)	ADCON2[2:0]		RW-000
ADFBEN	ADC 阈值比较结果匹配事件触发 PWM 故障刹车 1 = 使能 0 = 关闭	ADCON3[7]	0x186	RW-0
ADCMPOP	ADC 阈值比较的极性 1 = ADC 结果的高 8 位 < ADCMPH[7:0] 0 = ADC 结果的高 8 位 ≥ ADCMPH[7:0]	ADCON3[6]		RW-0
ADCMPEN	ADC 阈值比较 1 = 使能 0 = 关闭	ADCON3[5]		RW-0
LEBADT	LEB 结束后，ADC 开始自动转换 1 = 触发 ADC 转换 0 = 不触发 ADC 转换	ADCON3[3]		RW-0

表 1-2 ADC 相关用户寄存器

名称	地址	bit 7	bit 6	bit 5	bit 4	bit 3	Bit 2	bit 1	bit 0	复位值
ADRESL	0x111	A/D 转换结果低有效位								xxxx xxxx
ADRESH	0x112	A/D 转换结果高有效位								xxxx xxxx
ADCON0	0x113	-	CHS<3:0>				ADEX	GO/DONE	ADON	-000 0000
ADCON1	0x114	ADFM	ADCS<2:0>			ADNREF<1:0>		ADPREF<1:0>		0000 0000
ADCON2	0x115	ADINTREF<1:0>		ETGTYP<1:0>		ADDLY.8	ETGSEL<2:0>			0000 0000
ADDLY	0x188	ADDLY<7:0> / LEBPRL<7:0>								0000 0000
ADCON3	0x186	ADFBEN	ADCMPOP	ADCMPEN	-	LEBADT	-			000- 0---
ADCMPH	0x187	ADCMPH<7:0>								0000 0000
LEBCON	0x185	LEBEN	LEBCH		-	EDGS	-			000- 0---

表 1-3 ADC 相关用户寄存器地址

1.2. ADC 配置

配置 ADC 包括以下设置 (更改配置时需设置 ADON=0 以关闭 A/D 转换或外部触发)：

- 通道选择
- ADC 参考电压
- ADC 转换时钟源
- 转换结果格式
- 触发源
- ADC 延时或前沿消隐 (LEB)
- 响应 (中断设置)

通道选择 – 由 CHS 寄存器选择输入通道，连接到用于 ADC 转换的采样保持电路。相应的 I/O 需设置 TRISx = 1 和 ANSEL0x = 1 来配置成模拟输入。

ADC 参考电压 ($V_{ADC-REF}$) – ADC 以 2 个参考电压作为相对值来测量输入模拟电压：V_{REF}+ 和 V_{REF}−。参考电压可以选择为：

- V_{REF}+可选 VDD，V_{REF}−可选 GND
- 内部参考电压
- 内部参考电压加外部电容
- 外部参考电压 (V_{REF}+为 PA4, V_{REF}−为 PA5)

V_{REF}+和 V_{REF}−可以为上述选择的不同组合，但不可以同时选择内部参考电压，否则 V_{REF}−将强制连接到 GND。

内部参考电压可以为 0.5V，2.0V，3.0V，或 “未连接” (参阅 “ADINTREF”，**表 1-2**)。

ADC 转换时钟选择 – ADC 可通过指令选择 8 种时钟频率 (参阅 “ADCS”，**表 1-2**)：

- TSEL = 2T 时为 SysClk/N；TSEL = 4T 时为 SysClk/2N；N = 1, 2, 4, 8, 16, 32, 64
- LIRC (256 kHz 或 32 kHz)

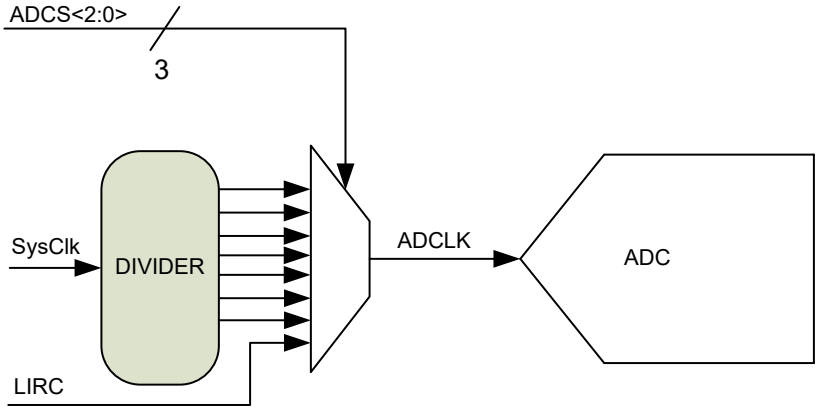

图 1-2 ADC 时钟配置

转换结果格式 – A/D 转换结果可储存为左对齐或右对齐两种格式(参阅 ”ADFM”，**表 1-2**)。

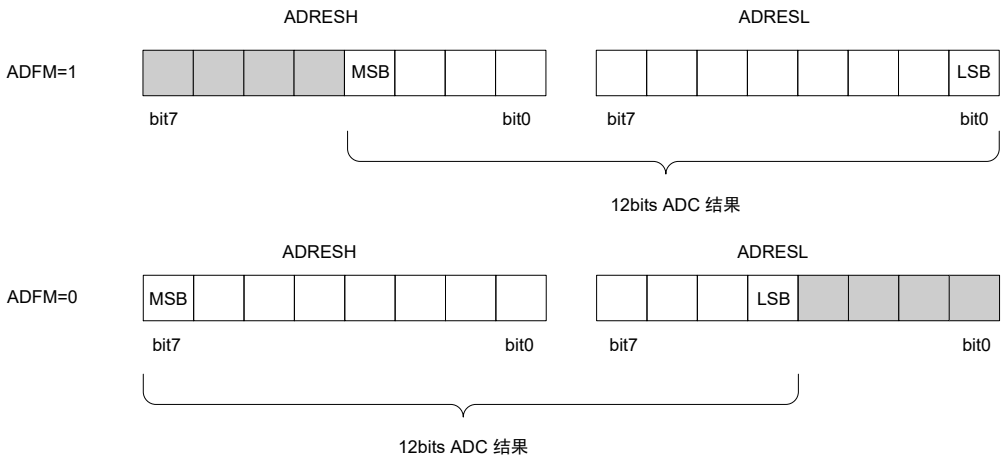

图 1-3　ADC 转换结果格式

1.2.1. ADC 触发和延时配置

ADC 转换可由指令(ADEX = 0)、PWM 边沿或 IO(PA4)转变沿(ADEX = 1)触发。其中，PWM 或 PA4 的触发沿可选择为 ”上升沿” 或 ”下降沿” (参阅 “ETGTYP”，**表 1-2**)。

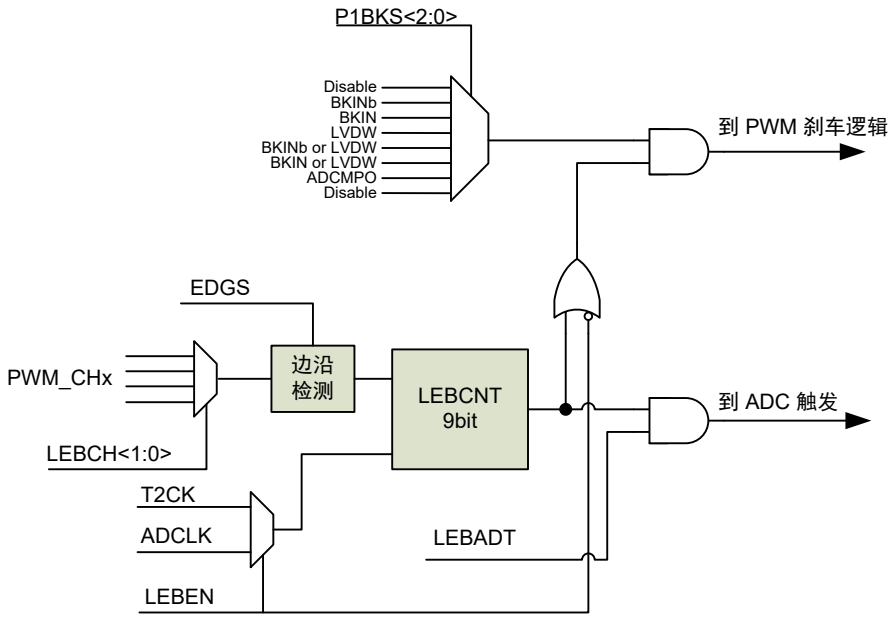

图 1-4　LEB 结构框图

在高速开关应用中，开关(如 MOSFETs/IGBTs)导通瞬间通常会产生极大的瞬变电流，而这些瞬变会导致测量误差。利用前沿消隐(LEB)功能，应用程序可忽略 PWM 输出边沿附近由 MOSFETs/IGBTs 开关所导致的预期瞬变。

LEB 和 PWM 的时钟源均为 T2CK(Timer2 时钟源)。LEB 计时期间，ADC 保持采样，直至 LEB 计时溢出(参阅 “LEBPR”，**表 1-2**)。在 LEB 计时周期内如果再次发生有效的 LEB 触发沿，则 LEB 定时器将清 0 并重新开始计数。

触发条件	延迟 / 消隐	触发通道
指令	(无延迟)	(N/A)
I/O (PA4)	(ADDLY+6) x T_{AD}; ADDLY = LEBPR	I/O (PA4)
PWM	(LEBPR+6) x T_{AD}	LEBEN = 0; ETGSEL (LEBCH 忽略)
	(LEBPR+4) x T_{T2CK} + 2 x T_{AD} (T_{T2CK} = Timer2 period)	LEBEN = 1; LEBCH (ETGSEL 忽略)

表 1-4 ADC 触发，延迟和通道设置

如果由软件触发(ADEX = 0)，GO/DONE 由指令置位后立即启动 A/D 转换。如果由 PA4 或 PWM 触发，则有一定的延迟时间(“6 x T_{AD}” 或 “4 x T_{T2CK} + 2 x T_{AD}”，参阅 **表 1-4**)。另外可通过设置 ADDLY/LEBPR 寄存器在 GO/DONE 置位前增加额外的延迟。ADC 延时定时器(ADDLY)和 LEB 定时器(LEBPR)共用同一个 9-bit 计数器，此计数器由 LEBPR9 和 LEBPRL[7:0]组成。延迟结束后采样保持电路将在 “0 – 1 x T_{AD}” 时间内断开。

注：

1. 在使能 LEB 前，需先设置 ADEX 和 ADON 寄存器。
2. ADC 转换完成前将忽略新的触发条件。.
3. 如果 LEBEN=1，则将忽略 ETGSEL，触发源即为 LEB 的触发源。此时由 LEB 定时器溢出触发 ADC 自动转换(参阅 “LEBADT”，**表 1-2**)。

1.2.2. ADC 中止转换

有时需中止 ADC 转换，比如需启动新的 ADC 采样。

- 当 ADEX = 0 (指令触发)时，可通过软件设置 GO/DONE = 0 来中止 ADC。
- 当 ADEX = 1 时，必须通过关闭 ADC 模块(ADON = 0)来中止 ADC。
- 当 ADC 转换被中止时，ADRESH 和 ADRESL 不会被更新，而是保持前一次的转换结果值。
- 系统复位时，由于相应的寄存器被复位，因此 ADC 将中止，且 ADC 模块被关闭。

1.2.3. 中断

ADC 模块在发生下列事件时将置位相应的中断标志位：

- ADC 转换完成 (ADCIF)
- ADC 阈值比较匹配 (ACMPIF)

每个中断模块均有其相应的中断使能位(ADCIE 和 ACMPIE)，和更高层级的外设总中断(PEIE)，以及最高级别的全局中断(GIE)。

无论中断使能位是否打开，发生中断事件时都将置位相应的中断标志位。是否触发中断和/或从睡眠中唤醒则取决于相应的使能控制位(GIE, PEIE, ADCIE 和 ACMPIE)。

注： ADC 转换完成后会自动将结果与 DCMPH 寄存器里的阈值进行比较(参阅“ADCMPEN”，**表 1-2**)。由 ADCMPOP 设置比较极性，当产生相应的匹配条件时将置位中断标志位 ACMPIF。仅转换结果的高 8 位用于阈值比较，因此 V_{REF}+和 V_{REF}−之间的比较 step 为 0.4%。当 ADFBEN = 1 时，也将使能相应中断。

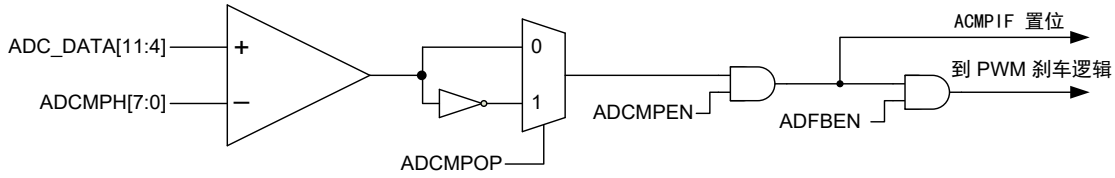

图 1-5 ADC 阈值比较结构框图

1.3. ADC 采样保持时间

采样保持时间 T_{ACQ}，必须足够长以保证内部 ADC 电压稳定在输入通道电压的 0.01%误差以内，以达到 12bit 的精度(0.024%)。采集时间和外部串联电阻的关系如下(**表 1-5**)：

$$T_{ACQ} > 0.09 \times (R + 1) \mu s;\ R \text{ 的单位为 } k\Omega.$$

当采样保持时间 T_{ACQ} 为 2μs 时，外部串联电阻必须≤ 21 kΩ。如果使用更大的串联电阻，则 T_{ACQ} 将成比例增加。结点漏电流限制了允许使用的最大串联电阻值。对于 5nA 的结点漏电流，在 50 kΩ的串联电阻上将产生 0.25mV (2V 参考电压的 0.0125%)的压降。而当温度超过 100°C 时，结点漏电流将大幅提高。因此，串联电阻越小越好。

串联电阻值	T_{ACQ}
> 50 kΩ	(不推荐)
43 kΩ	≥ 4.0 μs
21 kΩ	≥ 2.0 μs
< 21 kΩ	≥ 2.0 μs

表 1-5 不同的外部串联电阻与最短 T_{ACQ} 的对应关系

采样保持时间即为内部 ADC 观测输入通道电压的时间。

采样保持时间的开始=通道切换(参阅” CHS”)后或 ADC 稳定(参阅 T_{ST})后，以时间较迟者为准。

采样保持时间的结束 = 硬件触发延迟结束或软件 GO/DONE 置 1 后的 0 – 1 x T_{AD} 时间内，延迟时间由触发条件决定(参阅 **表 1-4**)，同时采样保持电路断开。

采样点 = 采样保持电路断开前的瞬间，有 0 – 1 x T_{AD} 时间的不确定性。

采样断开后开始数据转换，转换过程需 15 x T_{AD} 时间。因此从硬件触发延迟结束或软件 GO/DONE 置 1 后到数据转换完成需要 15 x T_{AD} 到 16 x T_{AD} 时间。数据转换完成后，采样保持电路重新闭合，开始下一个采样周期，同样需等待足够长的采样时间 T_{ACQ} 后，才能再次启动 A/D 转换。

1.4. ADC 最短采样时间

T_{AD} 为 ADC 的时钟周期。完整的 12-bit 转换所需最短时间：

T_{ACQ} + 16 T_{AD}

可保证真正 12-bit 精度的最高转换采样率为 48 kHz (或~21 μs/采样)。

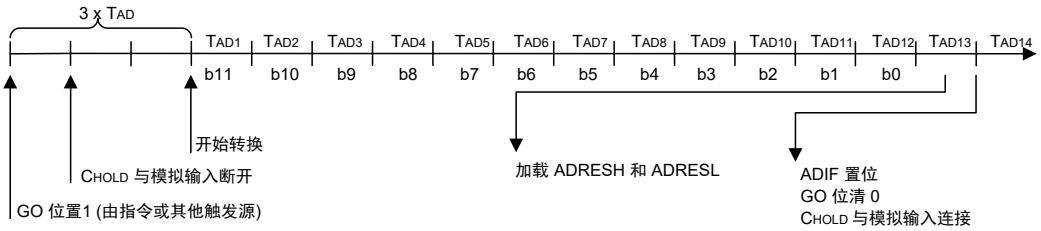

图 1-6 模数转换 T_{AD} 周期

1.5. ADC 转换步骤示例

设置 ADC：

1. 配置端口：
   a. 设置 TRISx = 1，禁止引脚输出驱动；
   b. 设置 ANSELx = 1，关闭数字输入、弱上拉和弱下拉功能；
2. 配置 ADC 模块：
   a. 选择 ADC 转换时钟源；
   b. 选择 ADC 参考电压；
   c. 选择 ADC 触发条件：软件、PA4-ADC_ETR 或 PWM，有或无 LEB；
   d. 选择转换结果格式；
   e. 使能阈值比较(可选)；
3. 配置 ADC 中断(可选)：
   a. 使能 ADC 和/或阈值比较中断；
   b. 使能外设总中断；
   c. 关闭全局中断(如需执行中断服务程序则使能)；
4. 打开 ADC 模块。然后等待所需 ADC 稳定时间 T_{ST} (~15 μs)，当 $V_{ADC-REF}$ 选择内部参考电压时，则需等待内部参考电压的稳定时间 T_{VRINT} 和 T_{ST} 时间的较长者，即 max(T_{VRINT}，T_{ST})。

至此，ADC 已准备好对不同的通道进行采样。对输入通道采样时：

1. ADC 输入选择为需测量的通道 (参阅"CHS")。
2. 如有必要，需清除 ADC 转换完成或阈值比较中断标志位。
3. 对采样时间 T_{ACQ} 有最低要求，T_{ACQ} 需足够长以保证内部 ADC 输入电容充满至输入通道电压的 0.01%误差以内。另外取决于触发类型，切换通道后或 ADC 稳定后(以时间较迟者为准)可能会有一定的延迟再触发。
   a. 对于软件触发，需要额外的 T_{ACQ} 时间。
   b. 对于 PA4-ADC_ETR 或 PWM 触发，除非使用非常大的串联电阻，否则内部延迟时间 (ADDLY+6) x T_{AD} 通常大于 T_{ACQ}，因此不需要再额外延迟 T_{ACQ}。
5. 等待所需的延迟后，由指令置位 GO/DONE，或等待硬件触发事件自动置位 GO/DONE，以启动 A/D 转换。

6. 通过以下方式等待 ADC 转换完成：
   a. 查询 GO/DONE 位；
   b. 等待 ADC 中断(使能中断时)；
7. 读取 ADC 转换结果。
8. 如有必要，清除 ADC 转换完成或阈值比较中断标志位。

注：

1. 虽然 GO/DONE 和 ADON 在同一个寄存器(ADCON0)中，但不应同时设置。
2. ADC 转换过程中或等待外部触发时，不可更改配置。建议在 ADON = 0 时进行更改。

以下为 ADC 程序示例 (输入采样通道为 PA0，ADC 时钟为 LIRC)：

```
BANKSEL ADCON1
LDWI B'01110000'              ; ADC LIRC clock
STR ADCON1
BANKSEL TRISA
BSR TRISA, 0                  ; Set PA0 to input
BANKSEL ANSEL0
BSR ANSEL0, 0                 ; Set-PA0 to analog
BANKSEL ADCON0
LDWI B'10000001'              ; Right justify,
STR ADCON0                    ; VDD, Vref, AN0, On
LCALL StableTime              ; ADC stable time
LCALL SampleTime              ; Acquisition delay, TACQ
BSR ADCON0, GO                ; Start conversion
BTSC ADCON0, GO               ; Conversion done?
LJUMP $-1                     ; No, test again
BANKSEL ADRESH;
LDR ADRESH, W                 ; Read upper 4 bits
STR RESULTHI                  ; Store in GPR space
BANKSEL ADRESL;
LDR ADRESL, W                 ; Read lower 8 bits
STR RESULTLO                  ; Store in GPR space
```

2 应用范例

```
/* 功能： 通过内部参考电压反推 VDD 电压
*  IC:      FT61F135   SOP-20
* 内部：   16M/2T
*
*                    FT61F135   SOP20
*                        ----------------
*  VDD----------|1(VDD)   (VSS)20|----------VSS
*  NC-----------|2(PC1)    (PA0)19|-----------NC
*  NC-----------|3(PC0)    (PA1)18|-----------NC
*  NC-----------|4(PB7)    (PA2)17|-----------NC
*  NC-----------|5(PB6)    (PA3)16|-----------NC
*  NC-----------|6(PB5)    (PA4)15|-----------NC
*  NC-----------|7(PB4)    (PA5)14|-----------NC
*  NC-----------|8(PB3)    (PA6)13|-----------NC
*  NC-----------|9(PB2)    (PA7)12|-----------NC
*  NC-----------|10(PB1)   (PB0)11|-----------NC
*                        ----------------
*/
#include "SYSCFG.h"
//宏定义*****************************************************
#define   unchar       unsigned char
#define   unint        unsigned int
#define   unlong       unsigned long

volatile unint   adcData;
volatile unint   VDD_Voltage;
/*-------------------------------------------------
 *  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/2T = 8MHz，0.125μs
    INTCON = 0B00000000;        //暂禁止所有中断

    PORTA = 0B00000000;
    TRISA = 0B00000000;         //PA 输入输出  0-输出  1-输入
    PORTB = 0B00000000;
    TRISB = 0B00000000;         //PB 输入输出  0-输出  1-输入
    PORTC = 0B00000000;
```

```
    TRISC = 0B00000000;          //PC 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
}
/*---------------------------------------------
 *  函数名：    DelayUs
 *  功能：      短延时函数  --16M-2T--大概快 1%左右.
 *  输入：      Time  延时时间长度  延时时长 Time Us
 *  返回：      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)
{
    ADCON1 = 0B10100101;         //右对齐，转换时钟 SysClk/32，负参考电压 GND
                                 //正参考电压 VDD
    //Bit7:ADFM A/D 转换结果格式
    //1 =  右对齐。
    //0 =  左对齐。

    //Bit6~Bit4:ADCS ADC 转换时钟源
    //TSEL=2T
    //000 = SysClk/2
    //001 = SysClk/8
    //010 = SysClk/32
    //011 = SysClk
    //100 = SysClk/4
    //101 = SysClk/16
    //110 = SysClk/64
    //111 = LIRC
```

```
//Bit3~Bit2:ADNREF ADC 负参考电压
//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:ADPREF ADC 正参考电压
//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;
//Bit6~Bit3:CHS ADC 模拟通道选择位
//0000 = AN0
//0001 = AN1
//0010 = AN2
//0011 = AN3
//0100 = AN4
//0101 = AN5
//0110 = AN6
//0111 = AN7
//1xxx = Int Vref（内部参考电压）

//Bit2:ADEX   ADC 触发条件
//0 = 由置位指令 GO/DONE（软件触发）
//1 = 由 PA4 或 PWM 置位 G0/DONE（硬件触发）

//Bit1:G0/DONE ADC 转换启动和状态位
//0 = A/D 转换完成/未进行转换
//1 = A/D 转换正在进行或硬件触发延时正在计数

//Bit0:ADON 使能 ADC
//0 = ADC 被禁止且不消耗工作电流
//1 = ADC 被使能

DelayUs(200);
DelayUs(250);                    //必须要延时 450us

ADCON2 = 0B01000000;             //选择内部正参考电压 2V，无外部触发源
//Bit7~Bit6:ADINTREF ADC 内部参考电压
//00 = 0.5V
//01 = 2.0V
```

```
    //10 = 3.0V
    //11 =  未连接

    //Bit5~Bit4:ETGTYP  外部触发沿（当 ADEX=1 时适用）
    //00 = PWM  或  ADC_ETR 脚的下降沿
    //01 = PWM  或  ADC_ETR 脚的上升沿

    //Bit3:ADDLY.8/LEBPR9 ADC 外部触发延时计数器或 LEB 计数器的第 8 位

    //Bit2~Bit0:ETGSEL  外部触发源（当 ADEX=1 时适用）
    //选择 PWM 源时需要配置 TIMER 为 PWM 输出模式并使能输出。
    //000 = P1A0
    //001 = P1A0N
    //010 = P1B
    //011 = P1C
    //100 = P1D
    //101 = ADC_ETR
    //11x =  无

    ADCON3 = 0B00000000;
    //Bit7:ADFBEN ADC 比较结果响应故障刹车使能
    //0 =  禁止
    //1 =  使能

    //Bit6:ADCMPOP ADC 比较器输出极性
    //0 = ADC 结果的高八位大于或等于 ADCMPH[7:0]
    //1 = ADC 结果的高八位小于 ADCMPH[7:0]

    //Bit5:ADCMPEN ADC 结果比较使能位
    //0 =  关闭
    //1 =  使能

    //Bit3: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

    ADCMPH = 0B00000000;          /ADC 比较阈值,仅 8 位，用于 ADC 结果高 8 位比较
    ADON=1;                       //使能 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)
 ------------------------------------------------*/
unint GET_ADC_DATA (unchar adcChannel)
{
    ADCON0 &= 0B00000111;
    ADCON0 |= adcChannel<<3;              //重新加载通道值
    DelayUs(40);                          //廷时等待电压稳定  Tst >10us
    ADCON0 = ADCON0|0x02;                 //启动 ADC
    NOP();
    NOP();
    while(ADCON0&0x02);                    //等待 ADC 转换完成

    return   (unint)(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(0x08);     //用 VDD 做基准测量内部参考通道  的 AD 值.
        adcData >>=4;                     //AD 值缩小 16 倍

        VDD_Voltage = (unlong)((512*100/adcData));
                                          //电压放大 100 倍，精确到 0.01V
                                          //VDD = (100*Vref*4095)/AD
        NOP();
        NOP();
    }
}
```

联系信息

Fremont Micro Devices Corporation

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

Tel: (+86 755) 8611 7811
Fax: (+86 755) 8611 7810

Fremont Micro Devices (HK) Limited

#16, 16/F, Block B, Veristrong Industrial Centre,
34−36 Au Pui Wan Street, Fotan, Shatin, Hong Kong SAR

Tel: (+852) 2781 1186
Fax: (+852) 2781 1144

http://www.fremontmicro.com