



## 概述

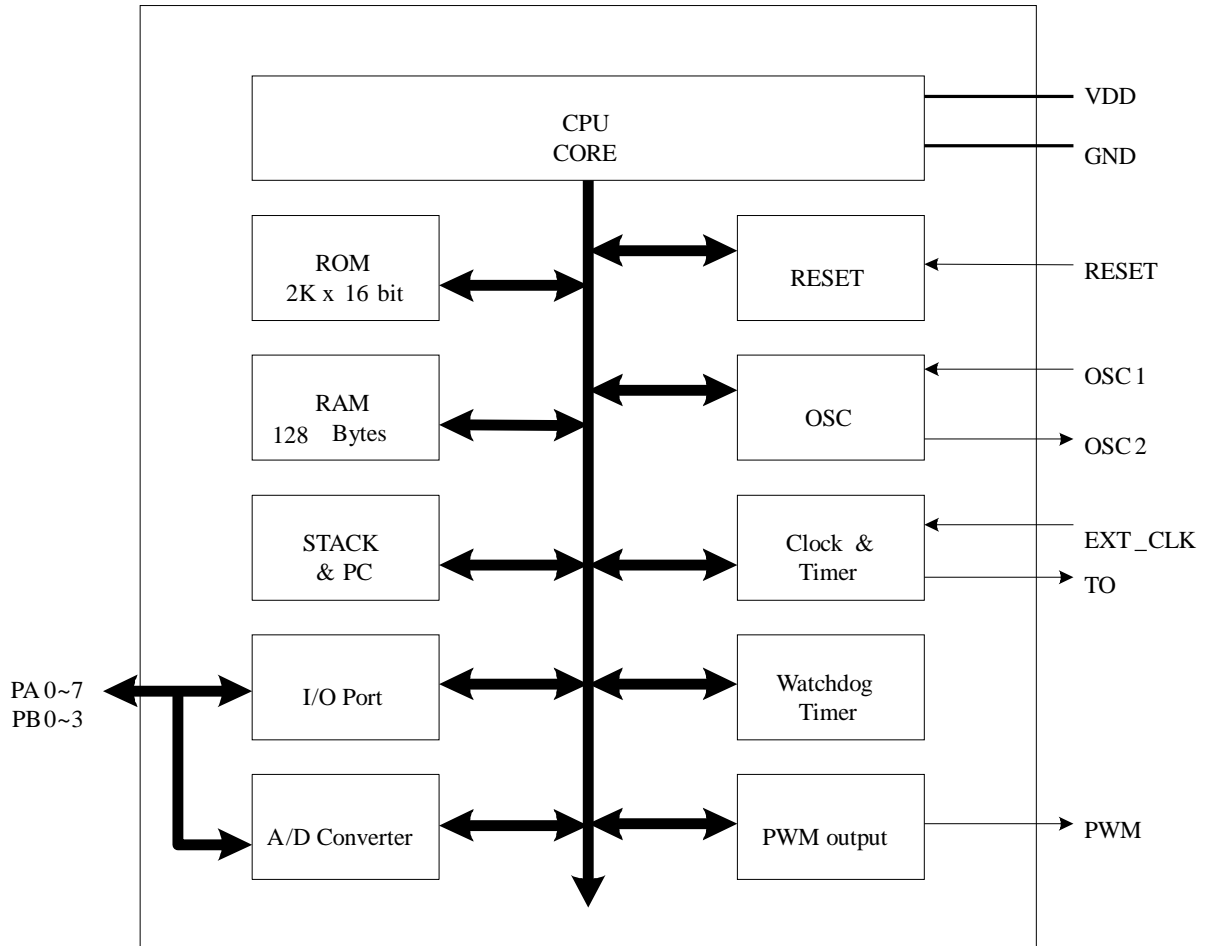
MK7A25P 是带 10 位 A/D 转换器的 RISC 高性能 8 位微控制器。它内部包含 2K 字节的一次性可编程只读存储器、128 字节数据存储器、定时器/计数器、捕捉、中断、LVR（低电压复位）、I/O 口和 PWM 输出。

### 1. 基本特性

- ROM: 2K×16 bits
- RAM: 37×8 bits (特殊寄存器) + 128×8 bits (通用寄存器)
- 堆栈: 8 级
- 一个指令周期由两个系统时钟组成
- 复位模式:
  - (a) 上电复位
  - (b) 低电压复位
  - (c) RESETB/PC1 (如果设置成复位脚位) 输入一个负脉冲
  - (d) 看门狗定时器计数溢出复位
- 双时钟模式
  - 外部 RC 或晶振振荡器
  - 内部 4MHz RC 振荡器
- 定时器/计数器
  - TM1: 16-bit, 捕捉 & 定时器
  - TM2: 8-bit, PWM (period) & 定时器
  - TM3: 8-bit, PWM (duty) & 定时器
  - TO: TM2 (PWM) 时钟输出
- 看门狗定时器: 芯片内 WTD 是基于一个内部 RC 振荡器(仅 WDT 使用)。有 8 个周期可供选择。使用者可通过使用预分频器来延长 WDT 溢出周期。
- 中断结果:
  - (a) 外部中断 (PA7~PA0)
  - (b) 内部定时器/结果计数器中断 (TM1~TM3)
  - (c) ADC结束转换中断
- I/O 口: 16 脚位
- PWM: 一个通道
- ADC: 最多 10 (6+4) -bit 及 4 通道, 至少 10-bit 精度。它能在转换模式或比较模式下使用
- 唤醒模式:
  - 1. A 口 (PA7~PA0) 脚位变化唤醒
  - 2. i\_WDT 唤醒
- 封装类型
  - MK7A25PD20C: 20 pin DIP
  - MK7A25PS20C: 20 pin SOP
  - MK7A25PD18C: 18 pin DIP
  - MK7A25PS18C: 18 pin SOP
  - MK7A25PD14C: 14 pin DIP
  - MK7A25PS14C: 14 pin SOP



## 2. 图表





### 3. 脚位定义 & 管脚分配

PA 3 / TO	1	●	20	PA 4 / EXT_CLK
PA 2	2		19	PA 5
PA 1	3		18	PA 6
PA 0	4		17	PA 7
PC 5	5		16	PC 4
PB 3 / AN 3	6		15	OSC2 / PC 3
PWM_OPT / PB 2 / AN 2	7		14	OSC1 / PC 2
PB 1 / AN 1	8		13	VDD
PB 0 / AN 0	9		12	RESETB / PC 1
VSS	10		11	PWM / PC 0

PDIP 20 or SOP20

PA 3 / TO	1	●	18	PA 4 / EXT_CLK
PA 2	2		17	PA 5
PA 1	3		16	PA 6
PA 0	4		15	PA 7
PB 3 / AN 3	5		14	OSC2 / PC 3
PWM_OPT / PB 2 / AN 2	6		13	OSC1 / PC 2
PB 1 / AN 1	7		12	VDD
PB 0 / AN 0	8		11	RESETB / PC 1
VSS	9		10	PWM / PC 0

PDIP 18 or SOP18

PA 3 / TO	1	●	14	PA 4 / EXT_CLK
PA 2	2		13	PA 5
PB 3 / AN 3	3		12	OSC 2 / PC 3
PWM_OPT / PB 2 / AN 2	4		11	OSC 1 / PC 2
PB 1 / AN 1	5		10	VDD
PB 0 / AN 0	6		9	RESETB / PC 1
VSS	7		8	PWM / PC 0

PDIP 14 or SOP14

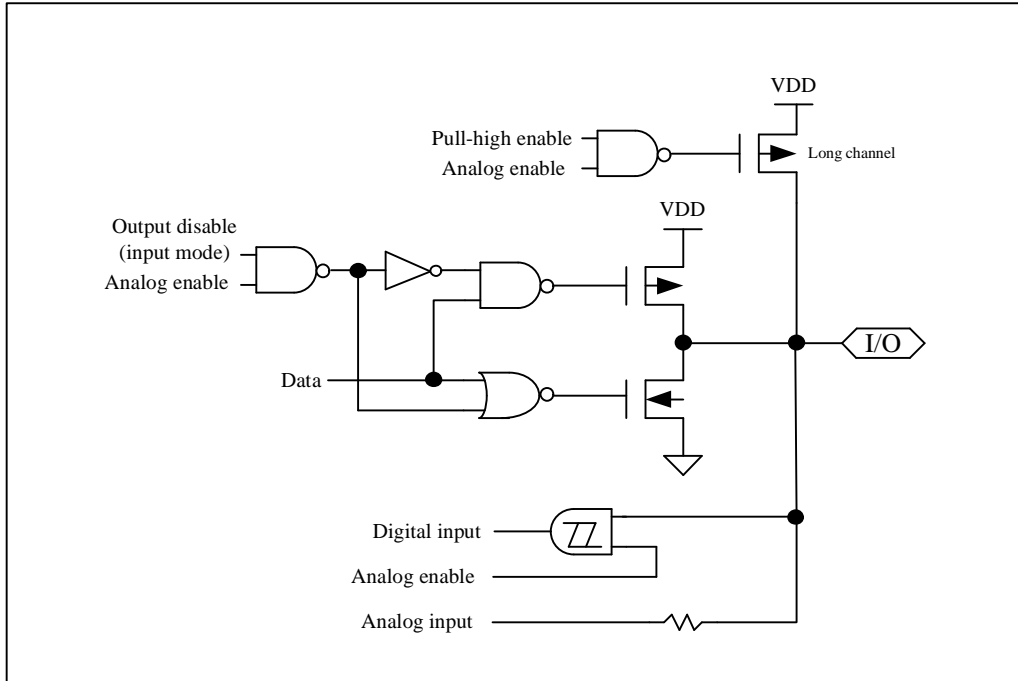


#### 4. 脚位说明

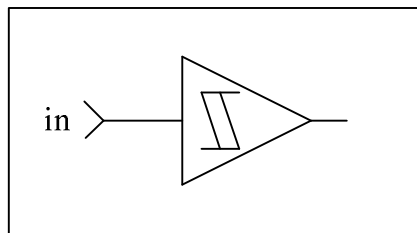
脚位名称	I/O	脚位类型	说明
PA0~2 PA5~7	I/O	D	1. I/O口（输入模式下会有上拉电阻） 2. 脚位改变时唤醒（选择） 3. 外部中断输入（选择）
PA3/TO	I/O	D	1. I/O口（输入模式下会有上拉电阻） 2. 脚位改变时唤醒（选择） 3. 外部中断输入（选择） 4. TO时钟输出
PA4/EXT_CLK	I/O	D	1. I/O口（输入模式下会有上拉电阻） 2. EXT_CLK时钟输入（或捕捉输入） 3. 脚位改变时唤醒（选择） 4. 外部中断输入（选择）
PB3/AN3	I/O	A	1. I/O口（输入模式下会有上拉电阻） 2. 模拟输入
PB1/AN1	I/O	A	
PB0/AN0	I/O	A	
PWM_OPT/PB2/AN 2	I/O	A	1. I/O口（输入模式下会有上拉电阻） 2. 模拟输入 3. 当PWM_OPT(\$1DH)设置为1，PWM选择输出
PWM/PC0	I/O	D	1. I/O口（输入模式下会有上拉电阻） 2. 当PWM_OPT(\$1DH)设置为0（初始值），PWM输出
RESETB/PC1	I	B	1. 复位脚位 2. 输入口
OSC1/PC2	I, I/O	C	1. 振荡器输入 2. I/O口（输入模式下会有上拉电阻）
OSC2/PC3	O, I/O	C	1. 振荡器输出 2. I/O口（输入模式下会有上拉电阻）
PC4~PC5	I/O	D	1. I/O口（输入模式下会有上拉电阻） 2. 初始状态上拉（UC5, UC4）使能
VDD	P		电源输入
VSS	P		接地输入



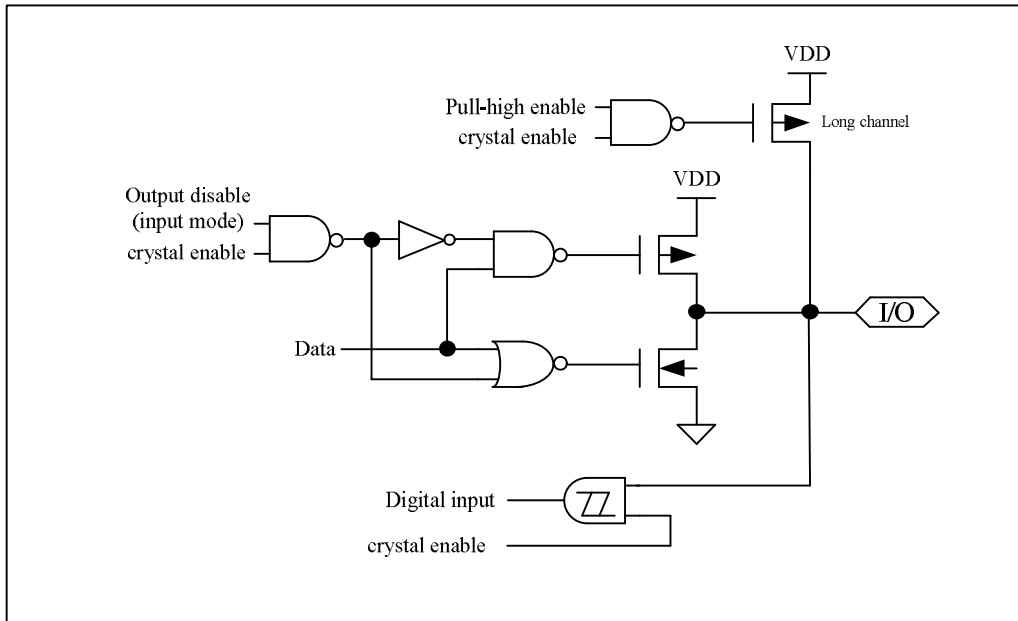
### 4.1 脚位电路



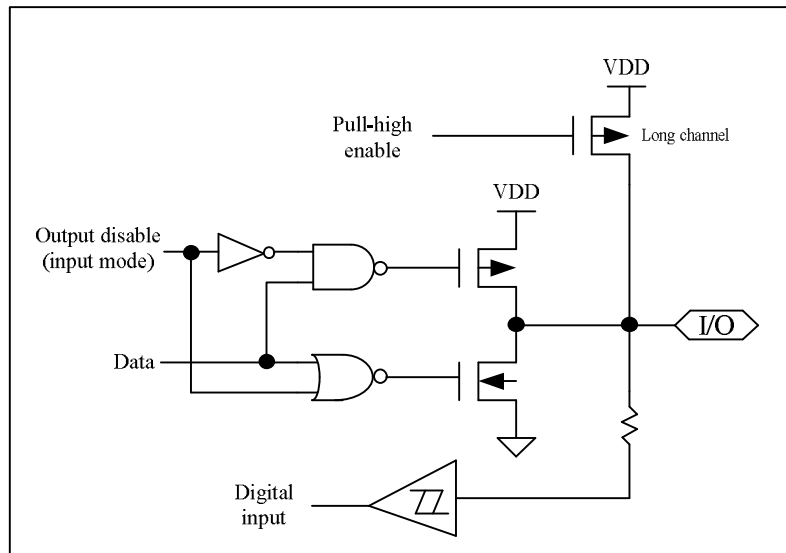
Pin circuit Type A



Pin circuit Type B



Pin circuit Type C



Pin circuit Type D



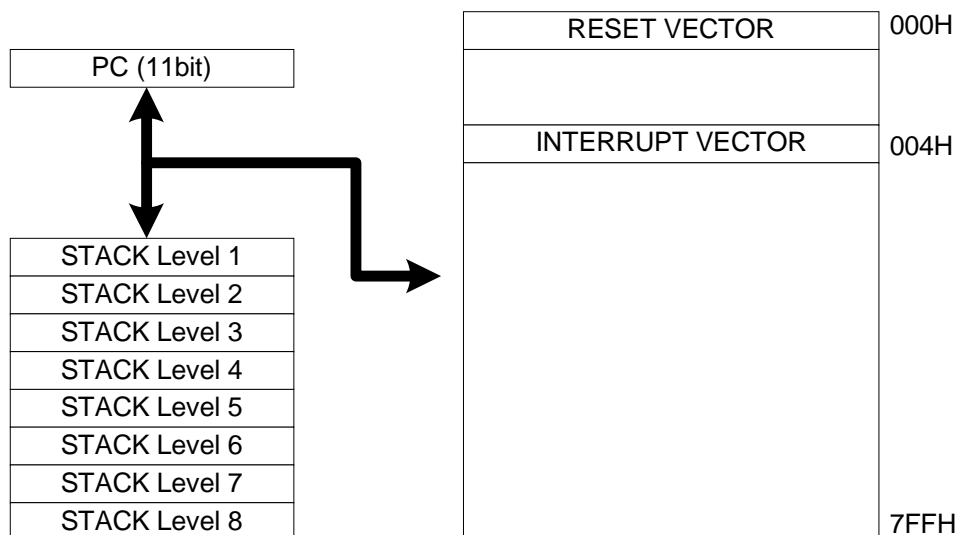
## 5. 存储器映象

MK7A25P 芯片带有两种存储器，分别是程序存储器（ROM）和数据存储器（RAM）。程序存储器用于存储程序、数据表及中断向量，它是连续的  $2048 \times 16\text{bits}$ ，不需要转换到 bank。数据存储器是  $165(37+128) \times 8\text{bits}$ ，它包括特殊功能寄存器和一般的数据存储器。

### 5.1 程序存储器（ROM）

指令和数据表存储在程序存储器内。程序存储器只能有一个中断向量存在，那意味着所有发生的中断都将跳到相同的向量。程序会通过中断标记来判断是哪一种中断发生。程序计数器（PC）有 11bit，它能直接寻找所有  $1024 \times 16\text{bits}$  位置地址。搜寻列表可以置于程序存储器的任何地方。

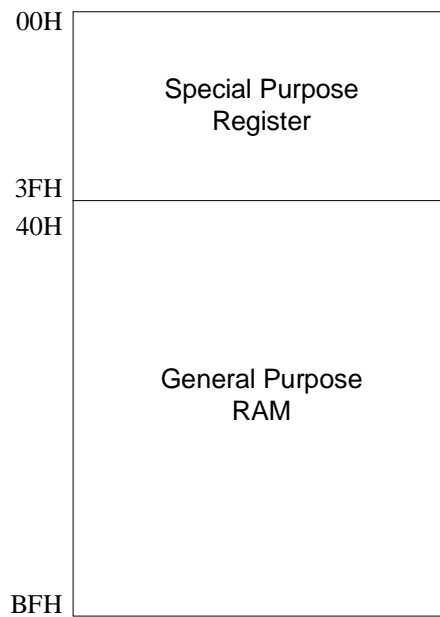
RESET 向量位于 000H，中断向量位于 004H。映象图如下所示：



### 5.2 数据存储器（RAM）

全部的数据存储器集都是  $165 \times 8\text{bits}$ ，它们包含两种寄存器组。一种是  $128 \times 8\text{bits}$  的通用存储器，另一种是  $37 \times 8\text{bits}$  的特殊寄存器。特殊寄存器的每一字节都用来存储控制数据和操作数据。

数据存储器映象如下所示：



### 5.2.1 特殊寄存器

Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CONFIG_L		RST_DEF	LV1	LV0	WDTE	CPRT	INRC	FOSC1	FOSC0
CONFIG_H		ADJ6	ADJ5	ADJ4	ADJ3	ADJ2	ADJ1	ADJ0	RTCEN
INDF	\$00	A7	A6	A5	A4	A3	A2	A1	A0
PCL	\$01	A7	A6	A5	A4	A3	A2	A1	A0
PCH	\$02	--	--	--	--	--	A10	A9	A8
STATUS	\$03	--	--	--	$\overline{TO}$	$\overline{PD}$	Z	DC	C
FSR	\$04	D7	D6	D5	D4	D3	D2	D1	D0
<b>I/O PAD &amp; Control</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_DIR	\$05	IOA7	IOA6	IOA5	IOA4	IOA3	CA2	IOA1	IOA0
PA_DAT	\$06	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PB_DIR	\$07	--	--	--	--	IOB3	IOB2	IOB1	IOB0
PB_DAT	\$08	--	--	--	--	IOB3	IOB2	IOB1	IOB0
PC_DIR	\$09	--	--	IOC5	IOC4	IOC3	IOC2	--	IOC0
PC_DAT	\$0A	--	--	DC5	DC4	DC3	DC2	DC1	DC0
<b>Timer 1: 16-bit (Timer &amp; capture)</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0





TM1_CTL1	\$13	TM1_EN	WR_CNT	SUR1	SUR0	EDGE	PRE2	PRE1	PRE0
TM1_CTL2	\$1F	E_CLR	--	--	--	--	--	--	--
CLR_CNT	\$21	CLR_CNT							
TM1L_LA	\$14	D7	D6	D5	D4	D3	D2	D1	D0
TM1H_LA	\$15	D7	D6	D5	D4	D3	D2	D1	D0
TM1L_CNT	\$16	D7	D6	D5	D4	D3	D2	D1	D0
TM1H_CNT	\$17	D7	D6	D5	D4	D3	D2	D1	D0
<b>Timer 2: 8-bit, PWM (period) &amp; Timer</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_CTL1	\$18	TM2_EN	WR_CNT	SUR1	SUR0	EDGE	PRE2	PRE1	PRE0
TM2_CTL2	\$19	MOD	PWM_OS	TO_E	--	POS3	POS2	POS1	POS0
TM2_LA	\$1A	D7	D6	D5	D4	D3	D2	D1	D0
TM2_CNT	\$1C	D7	D6	D5	D4	D3	D2	D1	D0
<b>Timer 3: 8-bit, PWM (duty) &amp; Timer</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM_OPT	\$1D	PWM_OPT	--	--	--	--	--	--	--
TM3_CTL	\$1E	TM3_EN	WR_CNT	SUR1	SUR0	EDGE	PRE2	PRE1	PRE0
TM3_LA	\$20	D7	D6	D5	D4	D3	D2	D1	D0
TM3_CNT	\$22	D7	D6	D5	D4	D3	D2	D1	D0
<b>IRQ</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRQM	\$25	INTM	ADCM	--	PAM	TM3M	TM2M/PWM	TM1M/CPT	--
IRQF	\$26	--	ADCF	--	PAF	TM3F	TM2F/PWM	TM1F/CPT	--
<b>ADC control</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AD_CTL1	\$29	EN	--	MODE	--	--	--	CHSEL1	CHSEL0
AD_CTL2	\$2A	RSUT	--	--	--	--	--	CKSEL1	CKSEL0
AD_CTL3	\$2B	--	--	--	--	--	PBSEL2	PBSEL1	PBSEL0
AD_DAT_L	\$2C	D1	D0	--	--	--	--	--	--
AD_DAT_H	\$2D	D9	D8	D7	D6	D5	D4	D3	D2
<b>Other</b>									
Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_PLU	\$31	UA7	UA6	UA5	UA4	UA3	UA2	UA1	UA0



PB_PLU	\$33	--	--	--	--	UB3	UB2	UB1	UB0
PC_PLU	\$35	--	--	UC5	UC4	UC3	UC2	--	UC0
PA_INTE	\$3A	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
WDT_CTL	\$3D	WDTEN	i_WDT	i_STAB	--	--	PRE2	PRE1	PRE0
TAB_BNK	\$3E	--	--	--	--	--	BNK2	BNK1	BNK0
SYS_CTL	\$3F	CLKS	--	--	--	--	--	STP1	STP0

< 注 > “—”：表示未使用

### 5.2.2 结构寄存器

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CONFIG_L	RST_DEF	LV1	LV0	WDTE	CPRT	INRC	FOSC1	FOSC0
-	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
CONFIG_H	ADJ6	ADJ5	ADJ4	ADJ3	ADJ2	ADJ1	ADJ0	EXT_CLK

- Bit15~9 (ADJ6~0)：用于校准内部RC振荡器
- Bit8 (EXT\_CLK)：EXT\_CLK 输入
  - 0：EXT\_CLK (PA4) 脚位是普通I/O脚位
  - 1：EXT\_CLK (PA4) 脚位是定时器源输入 & PA4输入
- Bit7 (RST\_DEF)：RESETB脚位定义
  - 0：RESETB是普通输入脚位
  - 1：RESETB是系统复位脚位
- Bit6~5 (LV1~0)：设置低电压复位 (LVR) 的复位电压级别

Bit6	Bit5	Detect voltage
LV1	LV0	
0	0	不使用
0	1	1.7V
1	0	2.0V
1	1	不使用

< 注 > 掉电电压会受到过程及温度的影响，因此列表中的电压会存在一些误差。

- Bit4 (WDTE)：看门狗定时器使能/禁止
  - 0：WDT 禁止
  - 1：WDT 使能
- Bit3 (CPRT)：ROM 密码保护位
  - 0：开
  - 1：关



- Bit2~0 (INRC, FOSC1~0): OSC 类型及系统时钟选择

Bit2	Bit1	Bit0	OSC Type	Resonance Frequency
INRC	FOSC1	FOSC0		
0	0	0	LS (low speed)	System clock=32~200KHz
0	0	1	NS (Normal speed)	System clock=200K~10MHz
0	1	0	HS (high speed)	System clock=10~20MHz
0	1	1	External RC	System clock=32K ~ 10MHz
1	0	0	LS & Internal RC	1. Dual clock mode LS & 4MHz 2. Initial system clock=4MHz
1	0	1	NS & Internal RC	1. Dual clock mode NS & 4MHz 2. Initial system clock=4MHz
1	1	0	HS & Internal RC	1. Dual clock mode HS & 4MHz 2. Initial system clock=4MHz
1	1	1	Internal RC	1. System clock=4MHz 2. OSC1 & OSC2 work as I/O ports

## 6. 功能描述

此芯片提供许多功能, 包括定时器, WDT, PWM, ADC, 捕捉, 中断, 数据表位置, 复位, 程序计数器及 STATUS 寄存器。我们将会在下面详细描述。

### 6.1 I/O口

该芯片有 3 个 I/O 口用于数据输入及输出, 每一个 I/O 口有不同的功能。A 口可通过选择寄存器实现外部中断, EXT\_CLK 时钟输入或捕捉输入。B 口具有 ADC 模拟输入功能。C 口带有外部 RC 振荡器输入, PWM 系统复位输入 (复位功能) 或输出功能。

#### 6.1.1 Port A

A 口有 3 个寄存器可设置 8 个 I/O 口, 分别是 PA\_DIR, PA\_DAT, PA\_PLU。A 口的每一脚位都可做为外部中断输入或一般 I/O 口。欲了解如何将这脚位设置为外部中断, 请参考章节 6.7。PA4 脚位带多功能, 使用者需先定义 8 位结构寄存器。

##### A. PA\_DIR(\$05H):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_DIR	IOA7	IOA6	IOA5	IOA4	IOA3	IOA2	IOA1	IOA0

- Bit7~0 (IOA7~0): 定义每一个脚位是输入口还是输出口

0: 输出

1: 输入



**B. PA\_DAT(\$06H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_DAT	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0

- Bit7~0 (DA7~0): 数据缓冲器

**C. PA\_PLU(\$31H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_PLU	UA7	UA6	UA5	UA4	UA3	UA2	UA1	UA0

- Bit7~0 (UA7~0): 上拉使能/禁止  
0: 上拉禁止  
1: 上拉使能

**6.1.2 B 口**

B 口有 3 个寄存器可设置 4 个 I/O 口，分别是 PB\_DIR, PB\_DAT, PB\_PLU。B 口的每一脚位都可设置为 ADC 模拟信号输入或普通 I/O 口。PB2/ PWM\_OPT 可被设置为 PWM 输出 (PC0 是普通 I/O 口)。欲了解如何使用 ADC 功能，请参考章节 6.8。当 B 口被设置为输入模式时，使用者可设置上拉。

**A. PB\_DIR(\$07H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB_DIR	--	--	--	--	IOB3	IOB2	IOB1	IOB0

- Bit3~0 (IOB3~0): 定义每一个脚位是输入口还是输出口  
0: 输出  
1: 输入

**B. PB\_DAT(\$08H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB_DAT	--	--	--	--	DB3	DB2	DB1	DB0

- Bit3~0 (DB3~0): 数据缓冲器

**C. PB\_PLU(\$33H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB_PLU	--	--	--	--	UB3	UB2	UB1	UB0

- Bit3~0 (UB3~0): 上拉使能/禁止  
0: 上拉禁止  
1: 上拉使能

**6.1.3 C 口**

C 口有 3 个寄存器可设置 4 个 I/O 口，分别是 PC\_DIR, PC\_DAT, PC\_PLU。PC0 (PWM) 可被设置为 PWM 输出 (PB2 是普通 I/O 口)。欲了解如何使用 PWM，请参考章节 6.3。当 PC3, PC2, PC0 被设置为输入模式时，使用者可设置上拉。CONFIG 寄存器可设置 PC1 (RESETB) 为系统复位 (低电压复位) 信号输入脚位。



通常情况下，PC2（OSC1）和 PC3（OSC2）是外部振荡器脚位，只有选择内部 RC 模式时，PC2 和 PC3 才是一般 I/O 口。

**A. PC\_DIR(\$09H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_DIR	--	--	IOC5	IOC4	IOC3	IOC2	--	IOC0

- Bit5, 4, 3, 2, 0 (IOC5, 4, 3, 2, 0): 定义每一个脚位是输入口还是输出口

0: 输出

1: 输入

< 注 > IOC1 仅是输入

**B. PC\_DAT(\$0AH):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_DAT	--	--	DC5	DC4	DC3	DC2	DC1	DC0

- Bit5, 4, 3, 2, 0 (DC5, 4, 3, 2, 0): 数据缓冲器

< 注 > DC1 仅是输入数据

**C. PC\_PLU(\$35H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_PLU	--	--	UC5	UC4	UC3	UC2	--	UC0

- Bit5, 4, 3, 2, 0 (UC5, 4, 3, 2, 0): 上拉使能/禁止

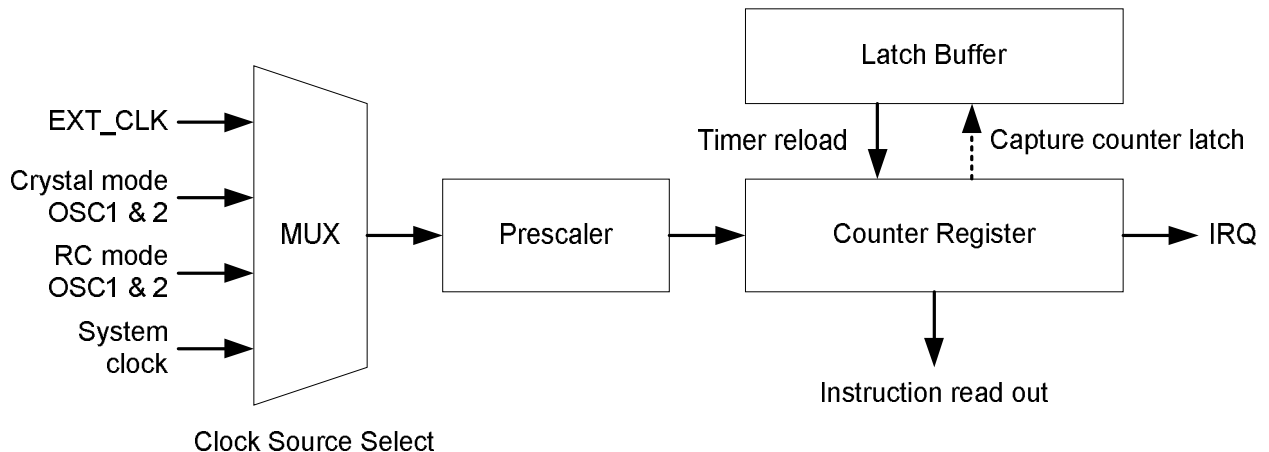
0: 上拉禁止

1: 上拉使能

- Bit5, 4 (UC5, 4): 在初始状态上拉使能

**6.2 定时器/结果计数器 (TM1, TM2, TM3)**

MK7A25P提供3个倒计时定时器/计数器和1个看门狗定时器。通过设置每一个定时器控制寄存器，计数器的时钟源可以是系统时钟，也可以是外部时钟。TM1是16位计数器，TM2和TM3是8位计数器。所有定时器带自动重复下载功能，TM1带捕捉功能，TM2/TM3组合后可实现PWM功能。寄存器详细设置及图表如下所示：





### 6.2.1 TM1

TM1 是 16 位定时器/计数器，有 5 个寄存器设置他的属性。欲了解如何使用 TM1 捕捉功能，请参考章节 6.4。

#### A. TM1\_CTL1 (\$13H):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1_CTL1	TM1_EN	WR_CNT	SUR1	SUR0	EDGE	PRE2	PRE1	PRE0

- Bit7 (TM1\_EN): 定时器1 (TM1) 使能/禁止
  - 0: TM1禁止
  - 1: TM1使能
- Bit6 (WR\_CNT): 锁存器数据写到计数器寄存器使能/禁止
  - 0: 锁存器数据写到计数器寄存器禁止
  - 1: 锁存器数据写到计数器寄存器使能

< 注 > 此位只有在新定时器/计数器数据的初始状态下被设置，才能让锁存器数据写到计数器寄存器。当定时器溢出，锁存器数据会自动重复下载到计数器寄存器。使用者不需要再设置一次，它与此位的状态无关。

- Bit5~4 (SUR1~0): TM1时钟源选择位

Bit5	Bit4	TM1时钟源
SUR1	SUR0	
0	0	EXT_CLK (PA4)
0	1	晶振模式 OSC1
1	0	RC 模式 (外部 & 内部 RC) OSC1
1	1	时钟源是系统时钟，捕捉输入是 PA4

< 注 > SUR1~0 定义 TM1 时钟源。如果 TM1 在捕捉功能下使用，则 SUR1~0 必须设置为 (1, 1)，这样就能记录从 PA4 脚位开始的数值，时钟源是系统时钟。如果捕捉正在运行，则 TM1 的计数器数据将会被锁定 TM1L\_LA 及 TM1H\_LA。

- Bit3 (EDGE): TM1时钟源边沿控制位

当TM1在定时器模式下使用

- 0: 时钟从低电平到高电平时，定时器加1
- 1: 时钟从高电平到低电平时，定时器加1

当TM1在捕捉模式下使用

- 0: 当EXT\_CLK (PA4) 从低电平到高电平时，保存TM1计数器寄存器到锁存器
- 1: 当EXT\_CLK (PA4) 从高电平到低电平时，保存TM1计数器寄存器到锁存器



- Bit2~0 (PRE2~0): 设置 TM1 预分频率 (定时器 & 捕捉)

Bit2	Bit1	Bit0	TMR1 Prescaler rate
PRE2	PRE1	PRE0	
0	0	0	1:1
0	0	1	1:2
0	1	0	1:4
0	1	1	1:8
1	0	0	1:16
1	0	1	1:32
1	1	0	1:64
1	1	1	1:128

#### B. TM1\_CTL2 (\$1FH):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1_CTL2	ENC	--	--	--	--	--	--	--

- Bit7 (TM1\_CTL1): 捕捉计数器自动清除
  - 0: 自动清除计数器 (硬体)
  - 1: 通过软体清除计数器 (设置CLR\_CNT)

#### C. CLR\_CNT (\$21H):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLR_CNT	CLR_CNT	--	--	--	--	--	--	--

- 写该寄存器来清除捕捉计数器，这是边沿触发，因而写0或1都是一样的。  
< 注 > 如果 TM1\_CTL2 Bit 7 被设置为 1，CLR\_CNT 没有写数据，则计数器将会保留不会被清 0。

#### D. TM1L\_LA/TM1H\_LA and TM1L\_CNT/TM1H\_CNT Register (\$14H, 15H, 16H, 17H)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1L_LA	D7	D6	D5	D4	D3	D2	D1	D0
TM1H_LA	D7	D6	D5	D4	D3	D2	D1	D0
TM1L_CNT	D7	D6	D5	D4	D3	D2	D1	D0
TM1H_CNT	D7	D6	D5	D4	D3	D2	D1	D0

- < 注 > TM1L\_CNT & TM1H\_CNT 两个寄存器为只读寄存器

#### 6.2.2 TM2 (或 PWM period)

TM2 是 8 位定时器/计数器，有 4 个寄存器设置他的属性。TM2 可当作 PWM period 使用，与 TM3 一起实现 PWM 波形。



**A. TM2\_CTL1 (\$18H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_CTL1	TM2_EN	WR_CNT	SUR1	SUR0	EDGE	PRE2	PRE1	PRE0

- Bit7 (TM2\_EN): 定时器2 (TM2) 使能/禁止  
0: TM2禁止  
1: TM2使能
- Bit6 (WR\_CNT): 锁存器数据写到计数器寄存器使能/禁止  
0: 锁存器数据写到计数器寄存器禁止  
1: 锁存器数据写到计数器寄存器使能

< 注 > 此位只有在新定时器/计数器数据的初始状态下被设置，才能让锁存器数据写到计数器寄存器。当定时器溢出，锁存器数据会自动重复下载到计数器寄存器。使用者不需要再设置一次，它与此位的状态无关。

- Bit5~4 (SUR1~0): TM2时钟源选择位

Bit5	Bit4	TM2时钟源
SUR1	SUR0	
0	0	EXT_CLK (PA4)
0	1	晶振模式 OSC1
1	0	RC 模式 (外部 & 内部 RC) OSC1
1	1	不使用

- Bit3 (EDGE): TM2时钟源边沿控制位  
0: 时钟从低电平到高电平时，定时器加1  
1: 时钟从高电平到低电平时，定时器加1
- Bit2~0 (PRE2~0): 设置TM2预分频率

Bit2	Bit1	Bit0	TM2 Prescaler rate
PRE2	PRE1	PRE0	
0	0	0	1:1
0	0	1	1:2
0	1	0	1:4
0	1	1	1:8
1	0	0	1:16
1	0	1	1:32
1	1	0	1:64
1	1	1	1:128





**B. TM2\_CTL2 (\$19H):**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_CTL2	MOD	PWM_OS	TO_E	--	POS3	POS2	POS1	POS0

- Bit7 (MOD): 模式选择位
  - 0: TM2在定时器模式下工作
  - 1: TM2在PWM模式下工作
- Bit6 (PWM\_OS): PWM输出状态选择位
  - 0: 初始输出状态是高电平, 当TM3定时器溢出时将变换为低电平
  - 1: 初始输出状态是低电平, 当TM3定时器溢出时将变换为高电平
- Bit5 (TO\_E): 定时器输出 (TO) 使能/禁止 (脚位与PA3共享)
  - 0: 设置该脚位为PA3一般I/O脚位
  - 1: 设置该脚位为TO (定时器输出脚位, 频率是TM2(PWM)计数器频率/2)

< 注 > 在 TO 信号输出前, PA3 必须设置为输出口

- Bit3~0 (POS3~0): PWM预分频率选择位 (仅在PWM模式下激活)

Bit3	Bit2	Bit1	Bit0	PWM Prescaler rate
POS3	POS2	POS1	POS0	
0	0	0	0	1:1
0	0	0	1	1:2
0	0	1	0	1:3
.	.	.	.	.
.	.	.	.	.
1	1	1	0	1:15
1	1	1	1	1:16

这些位控制当有多少次 PWM 波形输出将会出现一次 PWM 中断。

**C. TM2\_LA & TM2\_CNT (\$1AH, 1CH)**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_LA	D7	D6	D5	D4	D3	D2	D1	D0
TM2_CNT	D7	D6	D5	D4	D3	D2	D1	D0

< 注 > TM2\_CNT 寄存器为只读寄存器



### 6.2.3 TM3 (或 PWM duty)

TMR3 是 8 位定时器/计数器，有 4 个寄存器设置它的属性。TMR3 可当作 PWM duty 使用，与 TM2 共同控制实现 PWM 波形。

#### A. PWM\_OPT (\$1DH):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM_OPT	PWM_OPT	--	--	--	--	--	--	--

- Bit7 (PWM\_OPT): PWM输出脚位选择
  - 0: pwm从PC0 (初始) 输出
  - 1: pwm从PB2输出

#### B. TM3\_CTL (\$1EH):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM3_CTL	TMR3_EN	WR_CNT	SUR1	SUR0	EDGE	PRE2	PRE1	PRE0

- Bit7 (TM3\_EN): TMR3使能位
  - 0: TM3禁止
  - 1: TM3使能

< 注 > 当 TM2\_CTL2 被设置为 PWM 模式，此位将被抑制。TM3 就会成为 PWM 波形的占空比计数器。

- Bit6 (WR\_CNT): 锁存器数据写到计数器寄存器使能/禁止
  - 0: 锁存器数据写到计数器寄存器禁止
  - 1: 锁存器数据写到计数器寄存器使能

< 注 > 此位只有在新定时器计数器数据的初始状态下被设置，才能让锁存器数据写到计数器寄存器。当定时器溢出，锁存器数据会自动重复下载到计数器寄存器。使用者不需要再设置一次，它与此位的状态无关。

- Bit5~4 (SUR1~0): TM3时钟源选择位

Bit5	Bit4	TM3时钟源
SUR1	SUR0	
0	0	EXT_CLK (PA4)
0	1	晶振模式 OSC1
1	0	RC 模式 (外部及内部 RC) OSC1
1	1	不使用

- Bit3 (EDGE): TM3时钟源边沿控制位
  - 0: 时钟从低电平到高电平时，定时器加1
  - 1: 时钟从高电平到低电平时，定时器加1
- Bit2~0 (PRE2~0): 预分频器分配脚位



Bit2	Bit1	Bit0	TM3 Prescaler rate
PRE2	PRE1	PRE0	
0	0	0	1:1
0	0	1	1:2
0	1	0	1:4
0	1	1	1:8
1	0	0	1:16
1	0	1	1:32
1	1	0	1:64
1	1	1	1:128

### C. TM3\_LA & TM3\_CNT (\$20H, 22H)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM3_LA	D7	D6	D5	D4	D3	D2	D1	D0
TM3_CNT	D7	D6	D5	D4	D3	D2	D1	D0

< 注 > TM3\_CNT寄存器为只读寄存器

### 6.3 PWM（脉冲宽度调制）

PWM 波形由 TM2（period）和 TM3（duty）组合控制。通过设置寄存器，这两个定时器可用作通用定时器或 PWM 波形计数器。设置步骤如下例所示：

```

ORG      000H
LGOTO    MAIN          ; 跳到主程序
ORG      004H          ; 中断程序开始向量
BC       IRQF,b2       ; 清除PWM中断标记
                          ; 使用者的中断程序

                          IRETI

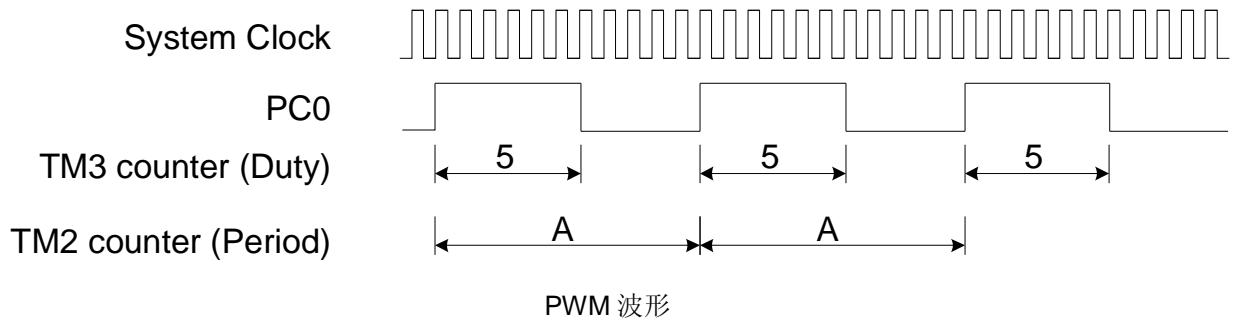
ORG      050H
MAIN:
MOVLA    B'XXXXXXXX0'  ; PC0（PWM）设置为输出
MOVAM    PC_DIR        ; 设置PC输出
MOVLA    B'01010000'
MOVAM    TM2_CTL1      ; 设置TM2为自动写，时钟源是晶振模式OSC1
MOVLA    B'10000000'
MOVAM    TM2_CTL2      ; 设置TM2为PWM模式，POS为1:1

```



```

MOVLA    0AH                ; period是0AH
MOVAM    TM2_LA             ; 设置TM2_LA=0AH, WR_CNT bit=1, 数据将写到TM2_CNT
MOVLA    B'01010000'
MOVAM    TM3_CTL           ; 设置TM3 WR_CNT=1, 时钟源是晶振模式OSC1
MOVLA    05H               ; duty是05H
MOVAM    TM3_LA            ; 设置TM3_LA =05H, WR_CNT bit=1, 数据将写到TM3_CNT
CLR      IRQF              ; 使能IRQM前, 清除IRQF
MOVLA    B'10000100'
MOVAM    IRQM              ; 使能INTM & TM2M中断
BC       TM2_CTL1,b6       ; TM2 WR_CNT=0
BC       TM3_CTL ,b6       ; TM3 WR_CNT=0
BS       TM2_CTL1,b7       ; 使能PWM, PC0开始输出
    
```





## 6.4 捕捉

捕捉功能提供波形测量，设置步骤如下所示：

- 使用者首先应设置 CONFIG 寄存器，设置 EXT\_CLK (bit 8) 为 “1”，使 PA4 为捕捉输入脚位。
- 设置 TM1 预分频率和捕捉条件（极性和边沿），然后使能捕捉中断。
- 捕捉中断发生，捕捉波形的长度将会被锁定在 TM1L\_LA 和 TM1H\_LA。

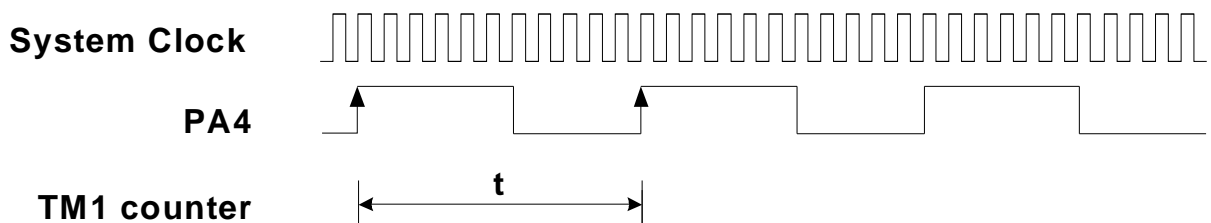
示例 1：捕捉一个周期（2 个上升沿）和使用中断（ENC=0，自动清除计数器）

```

ORG      000H
LGOTO    MAIN          ; 跳到主程序
ORG      004H          ; 中断程序开始向量
BC       IRQF,b1       ; 清除TM1中断标记
COM      TM1L_LA,a     ; 读低字节数据（FFH – 低寄存器数据）
MOVAM    40H           ; 保存数据到RAM
COM      TM1H_LA,a     ; 读高字节数据（FFH – 高寄存器数据）
MOVAM    41H           ; 保存数据到RAM
MOVVM    TM1H_LA,a
IRETI

ORG      050H
MAIN:
MOVLA    B'XXX1XXXX'
MOVAM    PA_DIR        ; 设置PA4输入
MOVLA    B'00110001'
MOVAM    TM1_CTL       ; 捕捉模式使能，TM1预分频率1:2.
CLR      IRQF          ; 使能IRQM前，清除IRQF
MOVLA    B'10000010'
MOVAM    IRQM          ; 中断功能使能，TM1中断使能
BS       TM1_CTL,b7    ; 开始捕捉功能

```





示例 2: 捕捉一个周期 (2 个上升沿), 不使用中断 (ENC=0, 自动清除计数器)

```
MOVLA    B'XXX1XXXX'  
MOVAM    PA_DIR                ; 设置PA4输入  
CLRM     IRQF  
MOVLA    B'10110000'  
MOVAM    TM1_CTL                ; 捕捉模式使能, TM1预分频率1:1  
Loop:    ; 等待捕捉标记使能循环  
BTMSC    IRQF,b1                ; 判断此位来知道标记被设置还是未被设置  
LGOTO    Cap_part                ; 如果标记是“1”则跳到读预下载寄存器  
LGOTO    Loop                    ; 如果标记是“0”则跳到Loop  
Cap_part: ; 捕捉后读预下载寄存器  
CLR      IRQF  
COM      TM1L_LA,a                ; 读低字节数据 (FFH – 低寄存器数据)  
MOVAM    40H                    ; 保存数据到RAM  
COM      TM1H_LA,a                ; 读高字节数据 (FFH – 高寄存器数据)  
MOVAM    41H                    ; 保存数据到RAM  
LGOTO    Loop
```

示例 3: 捕捉一个脉冲宽度及使用中断 (ENC=0, 自动清除寄存器)

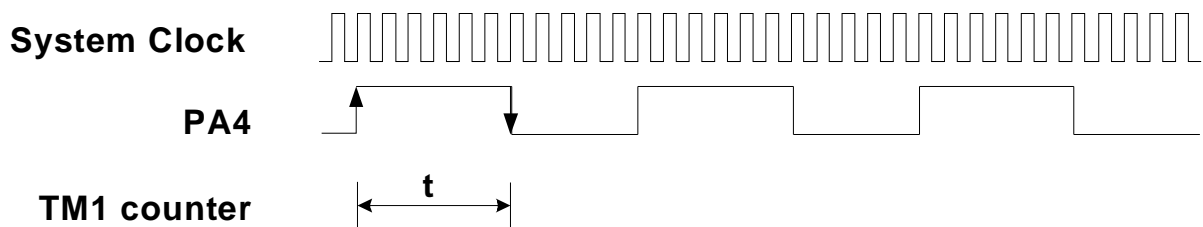
```
ORG      000H  
LGOTO    MAIN  
ORG      004H  
BC       IRQF,b1                ; 清除TM1中断标记  
BTSC     TM1_CTL,b3            ; 判断此位来知道哪一边沿发生  
LGOTO    Fall_edge            ; 如果现在是上升沿中断, 跳到上升_沿部分去设置控制寄存器  
                                         来调节上升_下降沿长度。如果现在是下降沿中断, 读数据  
  
BS       TM1_CTL,b3  
IRETI  
Fall_edge:  
BC       TM1_CTL,b3            ; PA4上升沿设置控制寄存器来捕捉中断发生。  
COM      TM1L_LA,a                ; 读低字节数据 (FFH – 低寄存器数据)  
MOVAM    40H                    ; 保存数据到RAM  
COM      TM1H_LA,a                ; 读高字节数据 (FFH – 高寄存器数据)  
MOVAM    41H                    ; 保存数据到RAM  
IRETI  
ORG      050H
```



```

MAIN:
MOVLA    B'XXX1XXXX'
MOVAM    PA_DIR                ; 设置PA4输入
MOVLA    B'00110000'
MOVAM    TM1_CTL                ; 捕捉模式使能。PA4上升沿捕捉中断服务。TM1预分频率1:1
CLR      IRQF                    ; 使能IRQM前，清除IRQF
MOVLA    B'10000010'
MOVAM    IRQM                    ; 中断功能使能，TM1中断使能
BS       TM1_CTL,b7              ; 使用者的程序

```



示例 4: 捕捉一个周期（1 个上升沿）及使用中断（ENC=1，通过软体清除计数器）

```

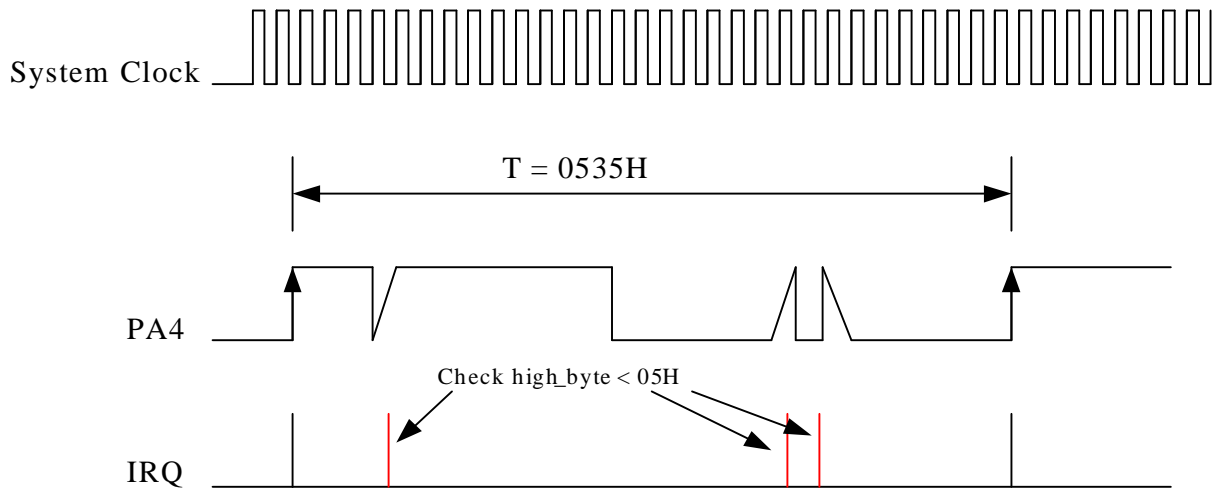
ORG      000H
LGOTO    MAIN                ; 跳到主程序
ORG      004H                ; 中断程序开始向量
BC       IRQF,b1              ; 清除TM1中断标记
MOV      TM1H_LA,a            ; 读高字节数据
SUB      OLD_HDATA,a          ; Old_hdata – New_hdata(a)
MOVAM    SUM_H                ; 保存高字节长度
SUB      LENGTH,a             ; 检查高字节长度=设置数值？（检查错误）
BTSS    STATUS,b2            ; Yes: 完成, No: 继续计数器
LGOTO    INTR
MOV      TM1L_LA,a            ; 读低字节数据
SUB      OLD_LDATA,a          ; Old_ldata – New_ldata(a)
MOVAM    SUM_L                ; 保存低字节长度
BTSS    STATUS,b0            ; 检查未有借出（C=1？）
DEC      SUM_H,m              ; 如果C=0（借出），SUM_H – 1
INTR:
        IRETI
ORG      050H

```



```

MAIN:
MOVLA    B'XXX1XXXX'
MOVAM    PA_DIR                ; 设置PA4输入
MOVLA    B'00110000'
MOVAM    TM1_CTL                ; 捕捉模式使能, TM1预分频率1:1
CLR      IRQF                    ; 使能IRQM前, 清除IRQF
MOVLA    B'10000010'
MOVAM    IRQM                    ; 中断功能使能前, TM1中断使能
MOV      CLR_CNT,m              ; 清除计数器
MOV      TM1L_LA,a
MOVAM    OLD_LDATA                ; 保存低字节数据到寄存器
MOV      TM1H_LA,a
MOVAM    OLD_HDATA                ; 保存高字节数据到寄存器
MOVLA    05H                    ; 设置捕捉长度=05
MOVAM    LENGTH                  ; 保存长度到寄存器
BS       TM1_CTL,b7              ; 开始捕捉功能
    
```







## 6.5 WDT (看门狗定时器)

WDT是防止软体故障及跳过含有不可预知结果的未知页面的定时器。WDT时钟源是一个独立的内部RC振荡器。此定时器会受温度、电压及不同产品批号的影响。

### A. WDT\_CTL (\$3DH):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WDT_CTL	WDTEN	i_WDT	i_STAB	--	--	PRE 2	PRE 1	PRE0

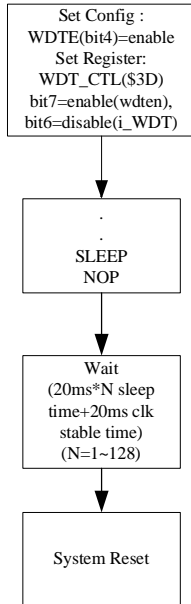
- Bit7 (WDTEN): 看门狗定时器使能位  
0: WDT禁止  
1: WDT使能
- Bit6 (i\_WDT): i\_WDT唤醒使能位, 在此模式, 脚位改变唤醒 (PA7) 禁止  
0: i\_WDT禁止  
1: i\_WDT使能
- Bit5 (i\_STAB): i\_STAB唤醒时间 (在i\_WDT模式) 设置位  
0: 唤醒时间=1.25ms (初始)  
1: 唤醒时间=625us
- Bit2~0 (PRE2~0): 设置预分频率。由于是RC OSC, 所有的数据不是精确数据

Bit2	Bit1	Bit0	WDT Prescaler rate
PRE2	PRE1	PRE1	
0	0	0	20mS
0	0	1	40mS
0	1	0	80mS
0	1	1	160mS
1	0	0	320mS
1	0	1	640mS
1	1	0	1.28S
1	1	1	2.56S

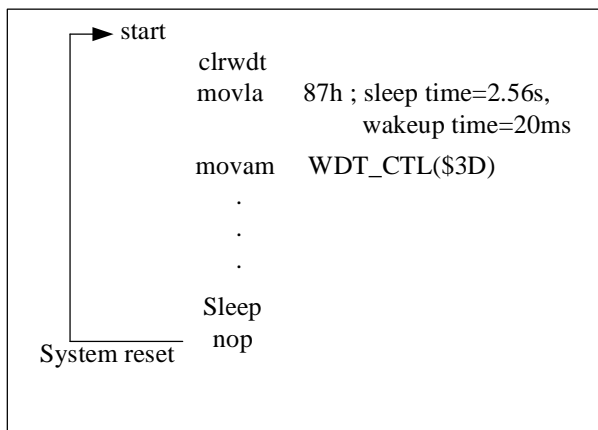
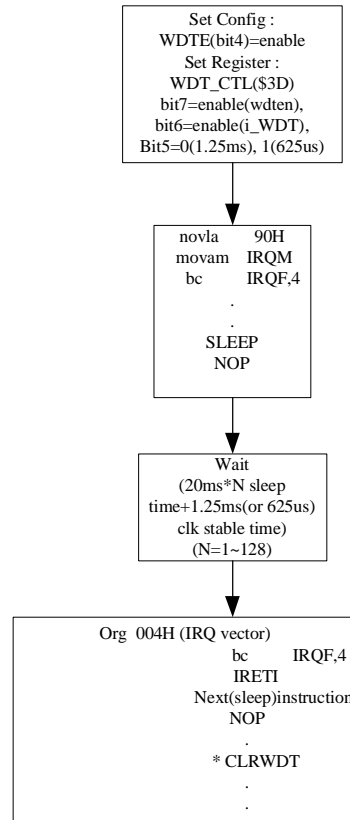
### 6.5.1 i\_WDT唤醒流程



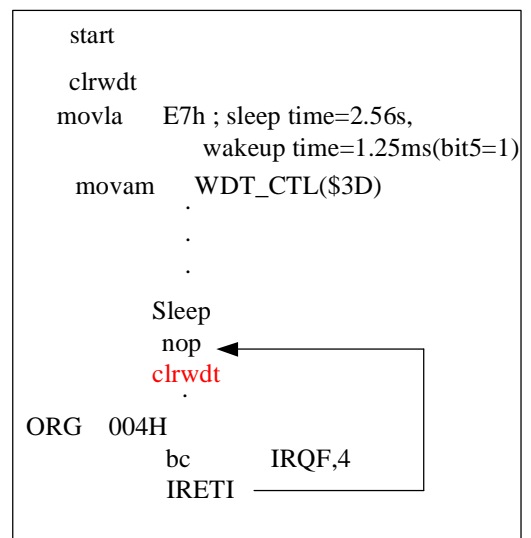
**WDT-Wakeup :**  
(Watch-dog timer wake-up)



**i\_WDT-Wakeup :**  
(Internal watch-dog timer  
wake-up)



\*When wakeup must  
CLRWDT, otherwise watch-  
dog timer will keep operation

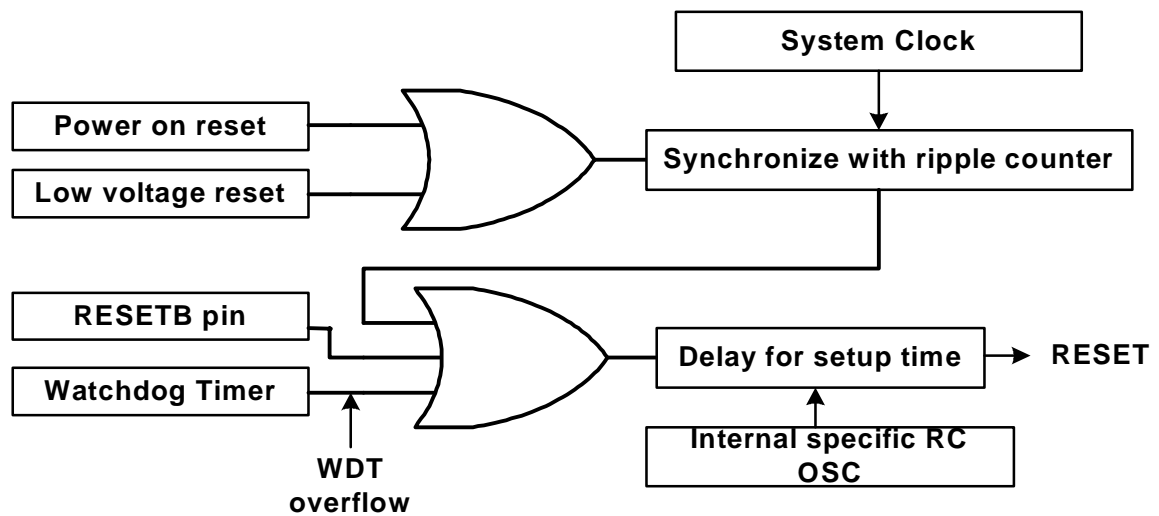




## 6.6 复位

以下列出了4种会引起复位的情况。掉电将会引起MK7A25P复位，检测电压依照CONFIG寄存器的bit6~bit5。这样能在供电不足的环境下保护芯片，最后两种情况我们称之为热复位。不同的复位都会影响寄存器和数据存储器。 $\overline{TO}$  和  $\overline{PD}$  位用来决定复位的类型。

- (1) 上电复位
- (2) 低电压复位 (LVR)
- (3) RESETB脚位复位 (输入一个负脉冲)
- (4) WDT定时器溢出复位



系统复位图表

< 注 > 看门狗设置时间为大约20ms，由于电源电压、进程及温度差异，在时间设置上会有一些偏差。

### 不同复位条件下的默认值

Address	Name	Cold Reset	Warm Reset
N/A	Accumulator	xxxx xxxx	pppp pppp
00H	INDF	0000 0000	0000 0000
01H	PCL	0000 0000	0000 0000
02H	PCH	---- -000	---- -000
03H	STATUS	0001 1xxx	0001 1xxx
04H	FSR	xxxx xxxx	pppp pppp
05H	PA_DIR	1111 1111	1111 1111
06H	PA_DAT	xxxx xxxx	pppp pppp
07H	PB_DIR	xxxx 1111	xxxx 1111



08H	PB_DAT	xxxx xxxx	xxxx pppp
09H	PC_DIR	xx11 11x1	xx11 11x1
0AH	PC_DAT	xxxx xxxx	xxpp ppxp
13H	TM1_CTL1	0000 0000	0000 0000
14H	TM1L_LA	1111 1111	1111 1111
15H	TM1H_LA	1111 1111	1111 1111
16H	TM1L_DAT	1111 1111	1111 1111
17H	TM1H_DAT	1111 1111	1111 1111
18H	TM2_CTL1	0000 0000	0000 0000
19H	TM2_CTL2	0000 0000	0000 0000
1AH	TM2_LA	1111 1111	1111 1111
1CH	TM2_DAT	1111 1111	1111 1111
1DH	PWM_OPT	0xxx xxxx	0xxx xxxx
1EH	TM3_CTL	0000 0000	0000 0000
1FH	TM1_CTL2	0000 0000	0000 0000
20H	TM3_LA	1111 1111	1111 1111
22H	TM3_DAT	1111 1111	1111 1111
25H	IRQM	00x0 000x	00x0 000x <sup>2</sup>
26H	IRQF	x0x0 000x	x0x0 000x
29H	AD_CTL1	0x0x xx00	0x0x xx00 <sup>2</sup>
2AH	AD_CTL2	0xxx xx00	0xxx xx00 <sup>2</sup>
2BH	AD_CTL3	xxxx x000	xxxx x000 <sup>2</sup>
2CH	AD_DAT_L	00xx xxxx	00xx xxxx
2DH	AD_DAT_H	0000 0000	0000 0000
31H	PA_PLU	0000 0000	0000 0000
33H	PB_PLU	0000 0000	0000 0000
35H	PC_PLU	0011 0000	0011 0000
3AH	PA_INTE	0000 0000	0000 0000
3DH	WDT_CTL	100x x111	100x x111
3EH	TAB_BNK	xxxx x000	xxxx x000
3FH	SYS_CTL	0xxx xx00	0xxx xx00

x: 未知的; p: 原来的数值; ? : 依据条件的数值; -: 不执行, 清“0”



## 6.7 中断

MK7A25P提供8种外部中断（PA0~7），3种内部定时器/结果计数器中断和1种A/D转换器中断。IRQM和IRQF寄存器用来控制或判断所有中断的请求状态。外部中断由PA0~7的信号键触发，相关的中断请求标记（PAF；IRQF的bit 4）将被设置。通过设置A/D转换器请求标记（ADCF；IRQF的bit 6），A/D转换器中断被初始化，通过结束A/D转换，中断发生。

IRQM用来使能/禁止中断，IRQF用来指出是哪一种中断发生。如果特殊IRQM不能使能则硬体中断将不会发生。但不管IRQM使能或禁止，IRQF都会有状态反应。例如，使用者使能TM1来开始计数，如果IRQM的bit 1使能，当定时器溢出，硬体中断将会发生，IRQF的bit 1将被设置，与此同时，程序将跳到中断向量。使用者应清除中断服务程序中的IRQF，否则中断将完全不工作。另一种情况是如果IRQM的bit 1禁止，当定时器溢出时，中断将不会产生，但IRQF的bit 1仍被设置，程序将不会跳到中断向量。

### A. IRQM (\$25H)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRQM	INTM	ADCM	--	PAM	TM3M	TM2M/PWM	TM1M/CAPT	--

- Bit7 (INTM): 通用使能脚位

0: 禁止，所有中断屏蔽

1: 使能，所有中断不屏蔽

当中断正在进行时，INTM将会被设置为“0”以防止其他中断的发生。当中断完成后，IRETI指令将会设置INTM为“1”。

- Bit6 (ADCM): ADC 结束转换 (EOC) 中断使能

0: 禁止中断

1: 使能中断

- Bit4 (PAM): PA 中断使能

0: 禁止中断

1: 使能中断

- Bit3 (TM3M): TM3 中断使能

0: 禁止中断

1: 使能中断

- Bit2 (TM2M/PWM): TM2/PWM 中断使能

0: 禁止中断

1: 使能中断

- Bit1 (TM1M/CAPT): TM1/捕捉中断使能

0: 禁止中断

1: 使能中断



**B. IRQF (\$26H)**

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRQF	--	ADCF	--	PAF	TM3F	TM2F/PWM	TM1F/CAPT	--

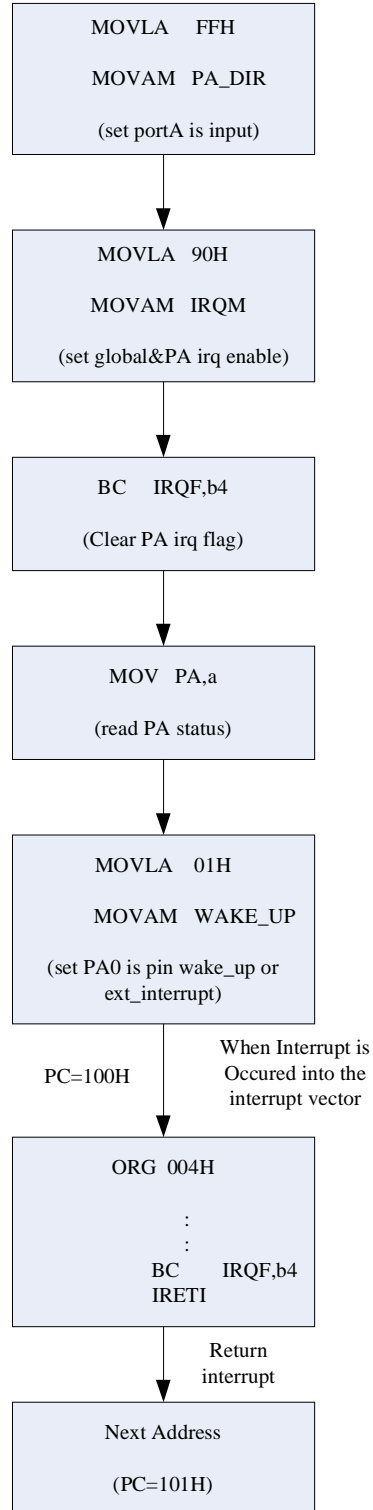
- Bit6 (ADCF): ADC 结束转换中断请求标记  
0: 结束转换中断请求关  
1: 结束转换中断请求开
- Bit4 (PAF): PA0~7 中断请求标记  
0: PA 中断请求关  
1: PA 中断请求开
- Bit3 (TM3F): TM3 中断请求标记  
0: TM3 溢出中断请求关  
1: TM3 溢出中断请求开
- Bit2 (TM2F/PWM): TM2/PWM 中断请求标记  
0: TM2 溢出中断请求关  
1: TM2 溢出中断请求开
- Bit1 (TM1F/CAPT): TM1/捕捉中断标记  
0: TM1 溢出或捕捉中断请求关  
1: TM1 溢出或捕捉中断请求开

**6.7.1 外部中断/唤醒功能**

A口 (PA) 提供外部中断和唤醒功能。当芯片不处于睡眠模式，PA输入信号将做为外部中断工作。当外部中断发生，程序将会跳到004H (中断向量)。如果芯片处于睡眠模式，PA输入信号将做为唤醒功能工作。当唤醒信号输入，芯片将会让系统时钟首先工作，然后等待唤醒定时器 (由WDT\_CTL寄存器\$3PH设置) 溢出，再之后，程序将跳到004H。以下流程图说明如何设置A口作为外部中断或唤醒功能工作。



PA Wake\_up & Ext\_interrupt :





## 6.8 ADC

MK7A21P提供4个通道和8位协议A/D转换器。A/D转换器包含5个寄存器，分别是AD\_CTL1 (29H)，AD\_CTL2 (2AH)，AD\_CTL3 (2BH)，AD\_DAT\_L (2CH) 及AD\_DAT\_H (2DH)。

### A. AD CTL1 (\$29H)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AD_CTL1	EN	--	MODE	--	--	--	CHSEL1	CHSEL0

- Bit7 (EN): ADC 使能位

0: ADC 禁止

1: ADC 使能

< 注 > 当结束转换，此脚位将会被设置为“0”

- Bit5 (MODE): ADC 模式选择位

0: ADC 通道作为 A/D 转换工作

1: ADC 通道作为比较仪工作

< 注 > (a) 如果此位是“1”，Vin 数据将会与 AD\_DAT 作比较，结果被存储在 AD\_CTL2 Bit7。如果此位是“0”，Vin 被转换成 8 位数位数据并保存在 AD\_DAT 寄存器。

(b) Vin: ADC 通道的输入电压

- Bit1~0 (CHSEL1~0): ADC 输入通道选择位

Bit1	Bit0	Input channel
CHSEL1	CHSEL0	
0	0	Channel 0, PB0
0	1	Channel 1, PB1
1	0	Channel 2, PB2
1	1	Channel 3, PB3

### B. AD\_CTL2 (\$2AH)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AD_CTL2	RSUT	--	--	--	--	--	CKSEL1	CKSEL0

- Bit7 (RSUT): 比较模式结果位

0:  $V_{in} < AD\_DAT$

1:  $V_{in} \geq AD\_DAT$

- Bit1~0 (CKSEL1~0): ADC 转换时钟源选择位





Bit1	Bit0	Conversion clock
CKSEL1	CKSEL0	
0	0	System clock X2
0	1	System clock X8
1	0	System clock X32
1	1	System clock X128

< 注 > 转换时钟决定转换率和精度。如果选择的是快速转换时钟，那将减少精度。如果使用者想得到更精确的 A/D 数据，推荐使用慢速。

### C. AD\_CTL3 (\$2BH)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AD_CTL3	--	--	--	--	--	PBSEL2	PBSEL1	PBSEL0

- Bit2~0 (PBSEL2~0): ADC 通道输入模式选择位

Bit2	Bit1	Bit0	PB3~PB0 configurations
PBSEL2	PBSEL1	PBSEL0	
0	0	0	PB3, PB2, PB1, PB0
0	0	1	PB3, PB2, PB1, AN0
0	1	0	PB3, PB2, AN1, AN0
0	1	1	PB3, AN2, AN1, AN0
1	X	X	AN3, AN2, AN1, AN0

< 注 > 最小化能耗，所有 I/O 脚在进入睡眠模式前都会被小心控制

### D. AD\_DAT\_L (2CH)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AD_DAT_L	D1	D0	--	--	--	--	--	--

### E. AD\_DAT\_H (2DH)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AD_DAT_H	D9	D8	D7	D6	D5	D4	D3	D2

< 注 > 此寄存器有两种不同的用法，如果在比较模式下工作，这些数据将会与 ADC 通道输入电压相比较。在 ADC 模式下，寄存器存储 ADC 转换数据。



示例:

```
LOOP1
    BTSC    AD_CTL1,b7 ; 确认A/D是否完成?
    BC     IRQF,b6 ; 清除A/D标记 ;
    MOVLA  03H
    MOVAM  AD_CTL1 ; set 设置PB3为A/D输入 ;
    MOVAM  AD_CTL2 ; 设置A/D clk=sys_clk/128;
    MOVLA  04H
    MOVAM  AD_CTL3 ; 设置PB0~PB3为模拟输入 ;
    BS     AD_CTL1,b7 ; 开始A/D转换 ;

CHECK_FLAG
    BTSS   IRQF,b6 ; 确认A/D标记是否等于1?
    LGOTO  CHECK_FLAG
    MOVLA  AD_DAT,a
    MOVAM  40H ; 读A/D数据到RAM
```

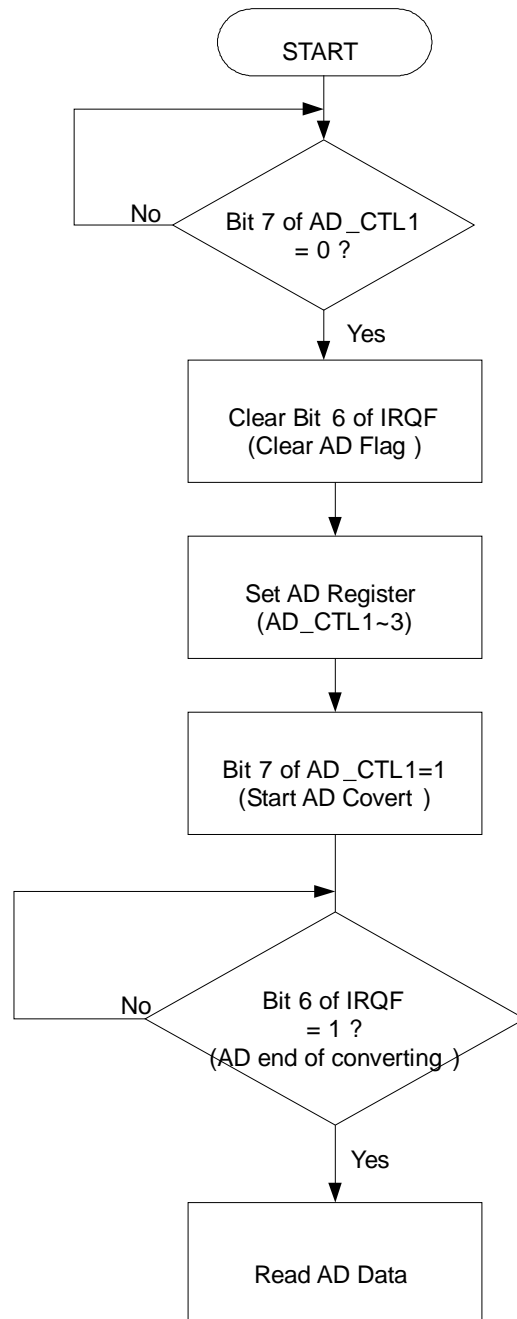


图. ADC设置流程



## 6.9 表格查询功能

MK7A25P提供表格查询功能。查询表格可以置于ROM空间的任何位置。TABRDL指令是读ROM表格的低字节。TABRDH是读高字节。TAB\_BNK寄存器用于定义表格位置（3+8=11bits-address bit， $2^{11}$ =2Kbytes-data byte）的高位（MSB）地址。

### 6.9.1 TAB\_BNK (\$3EH)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TAB_BNK	--	--	--	--	--	BANK2	BANK1	BANK0

- Bit2~0（BANK2~0）：高字节表格位置选择位

Bit2	Bit1	Bit0	BANK select
BANK2	BANK1	BANK0	
0	0	0	000 XXXX XXXX Table location
0	0	1	001 XXXX XXXX Table location
0	1	0	010 XXXX XXXX Table location
.	.	.	.
1	1	1	111 XXXX XXXX Table location

### 6.9.2 表格查询举例程序

以下程序的主要功能是表格查询，TABRDL和TABRDH的结果将会是55H和AAH（地址是0704H）。

```
#DEFINE    TAB_BNK    3EH           ; 定义地址，RAM 的 3EH 定义为 TAB_BNK

BUFA      EQU        43H           ; 定义地址，RAM 的 43H 定义为 BUFA

(address)  ORG        0700H        ; 程序从 ROM 的 0700H 开始
0700H     MOVLA      00H           ; 保存 00H 到 A 寄存器
0701H     DW         1122H        ; 存储 1122H 在 ROM 的 0701H
0702H     DW         3344H        ; 存储 3344H 在 ROM 的 0702H
0703H     DW         6677H        ; 存储 6677H 在 ROM 的 0703H
0704H     DW         55AAH        ; 存储 55AAH 在 ROM 的 0704H

          MOVLA      04H           ; 保存 04H 到 A 寄存器（低位地址）
          MOVAM      BUFA         ; 保存 A 寄存器的数值到 BUFA
          MOVLA      0FH           ; 保存 0FH 到 A 寄存器（高位地址）
          MOVAM      TAB_BNK      ; 保存 A 寄存器的数值到 TAB_BNK
          TABRDL     BUFA         ; 查询 TAB_BNK 的低字节数值和 BUFA 指定地址，
                                ; 保存到 A 寄存器
```



TABRDH      BUFA      ; 查询 TAB\_BNK 的高字节数值和 BUFA 指定地址，  
保存到 A 寄存器

## 6.10 系统控制

MK7A25P提供Auto-Bank功能和双重时钟操作模式。当序列管理跳跃，系统将自动保存PC的高字节，以防止PC进位溢出错误，那就是Auto-Bank的意思。双重时钟模式有内部RC和外部晶振时钟源。使用者可以在同一时间使用两种模式。例如，内部RC（4MHz）常用作系统时钟源，外部晶振（32KHz）用作计数器时钟源，因为外部晶振非常精确，使用者可以获得一个非常精确的定时器。

### 6.10.1 SYS\_CTL (\$3FH)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SYS_CTL	CLKS	--	--	--	--	--	STP1	STP0

- Bit7 (CLKS): 系统时钟源选择位，只有在双重时钟模式下激活

双重时钟模式：

0: 系统时钟是内部RC

1: 系统时钟是OSC（晶振）

单时钟模式：

不使用

- Bit1 (STP1): 内部RC振荡器控制位

双重时钟模式：

0: RC振荡开

1: RC振荡关

其他模式：

不使用

< 注 > 在设置 RC 振荡关之前，首先将 CLKS 转换到 OSC 振荡

- Bit0 (STP0): OSC（晶振）振荡器控制位

双重时钟模式：

0: OSC振荡开

1: OSC振荡关

其他模式：

不使用

< 注 > 在设置 OSC 振荡关之前，首先将 CLKS 转换到 RC 振荡



## 6.11 程序计数器 - PC

MK7A25P有一个11-bits程序计数器（PC），包含PCL（8-bits）和PCH（3bits）。PC被存储在程序通道内。如果使用者改变PCL的数值，程序将会跳到指示页面。

Ex1: PCH=01H, PCL=02H+10H=12H, 程序将跳到PC=112H

Ex2: PCH=01H, PCL=F0H+30H=20H带进位1, 程序将跳到PC=220H, 但PCH仍旧是01H

< 注 > (a) 当执行 RET 和 RETI, PCH 数据将不会被更新

(b) 当执行 LGOTO, LCALL, JZ, JC 和 RET, PCH 将会在精确的操作后被更新

### 示例 1:

以下程序说明 PCL 和 PCH 如何直接精确的工作。

```
#DEFINE      PCL          01H          ; 定义地址, RAM 的 01H 定义为 PCL
#DEFINE      PCH          02H          ; 定义地址, RAM 的 02H 定义为 PCH
#DEFINE      ADMIN        41H          ; 辅助 PCL 操作
(address)
1C0H         MOVLA        HIGH P1      ; 保存 P1 (1C6H) 高字节地址到 A 寄存器
; PC=1C0H, PCL=C0H, PCH=00H
1C1H         MOVAM        PCH          ; 保存 A 寄存器到 PCH (为了避免 PCL 操作的跳跃
; 错误, 首先存储真正的跳跃高字节地址到 PCH)
; PC=1C1H, PC =C1H, PCH=01H
1C2H         MOVLA        4BH          ; 保存 4BH 到 A 寄存器 (ADDAM PCL, M 的地址
; 是 1C5H, 准备跳到 210H, PCL 增加 210H-1C5H
; =4BH)
; PC=1C2H, PC =C2H, PCH=01H
1C3H         MOVAM        ADMIN        ; 准备 PCL 操作
; PC=1C3H, PCL=C3H, PCH=01H
1C4H         DEC          ADMIN, a     ; ADMIN-1 (真正的跳跃发生在 1C6H, 不在 1C5H,
; 因此 1C6H+(4BH-1H)=210H)
; PC=1C4H, PCL=C4H, PCH=01H
1C5H         ADD          PCL, M       ; PCL 用 A 寄存器增加, 结果存储在 PCL
; PC=1C5H, PCL=C5H, PCH=01H
P1:1C6H     NOP            ; 跳到 0210H
; PC=1C6H, PCL=C6+4AH=10H 带进位 1, 进位将
; 用 PCH 计数, PCH=01H, 用途 PC 高字节地址将是
; PCH+PCL's 进位=02H。程序将跳到 PC=210H
210H        MOVLA        00H          ; 用途功能部分
; PC=210H, PCL=10H, PCH=01H
```



**示例 2:**

以下程序说明 PCL 和 PCH 如何直接精确的工作。

(address)			
1C0H	MOVLA	03H	; 保存 <b>03H</b> 到 A 寄存器 ; <b>PC=1C0H, PCL=C0H, PCH=00H</b>
1C1H	MOVAM	PCH	; 保存 A 寄存器到 PCH ; <b>PC=1C1H, PCL=C1H, PCH=03H</b>
1C2H	MOVLA	4BH	; 保存 4BH 到 A 寄存器 ; <b>PC=1C2H, PCL=C2H, PCH=03H</b>
1C3H	MOVAM	ADMIN	; 准备 PCL 操作 ; <b>PC=1C3H, PCL=C3H, PCH=01H</b>
1C4H	DEC	ADMIN, a	; ADMIN-1 (真正的跳跃发生在 1C6H, 不在 1C5H, 因此 1C6H+(4BH-1H)=210H) ; <b>PC=1C4H, PCL=C4H, PCH=03H</b>
1C5H	ADD	PCL,M	; PCL 用 A 寄存器增加, 结果存储在 PCL ; <b>PC=1C5H, PCL=C5H, PCH=03H</b>
1C6H	NOP		; 跳到 410H ; <b>PC=1C6H, PCL=C6+4AH=10H 带进位 1, 进位将 用 PCH 计数, PCH=03H, 用途 PC 高字节地址将是 PCH+PCL's 进位=04H。程序将跳到 PC=410H</b>
410H	MOVLA	00H	; 用途功能部分 ; <b>PC=410H, PCL=10H, PCH=03H</b>

**示例 3:**

以下程序说明 PCL 和 PCH 如何通过 A 寄存器工作。

(address)			
018H	MOVLA	02H	; 保存 <b>02H</b> 到 A 寄存器
019H	MOVAM	PCH	; 保存 A 寄存器到 PCH (用途地址是 200H, 因此存 储 “02H” 到 PCH)
01AH	MOVLA	00H	; 保存 <b>00H</b> 到 A 寄存器
01CH	MOVAM	PCL	; 保存 A 寄存器到 PCL (用途地址是 200H, 因此存 储 “00H” 到 PCL)
01DH	NOP		; 跳到 200H
200H	MOVLA	00H	; 用途功能部分



## 6.12 STATUS寄存器

STATUS寄存器是一个包含零标记 (Z)，进位标记 (C)，四位进位标记 (DC)，掉电标记 ( $\overline{PD}$ )，看门狗定时器溢出标记 ( $\overline{TO}$ ) 的8位寄存器，它用于记录状态信息。

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	--	--	--	$\overline{TO}$	$\overline{PD}$	Z	DC	C

- Bit4 ( $\overline{TO}$ ): 定时器溢出标记位
- Bit3 ( $\overline{PD}$ ): 掉电标记位

$\overline{TO}$	$\overline{PD}$	说明
0	0	在睡眠模式中 WDT 定时器溢出
0	1	在普通模式中 WDT 定时器溢出
1	0	在睡眠模式中给 RESETB 输入一个“低电平”
1	1	上电复位
Unchanged	Unchanged	在普通模式中给 RESETB 输入一个“低电平”

- Bit2 (Z): 零标记位
  - 0: 逻辑操作结果不是0
  - 1: 逻辑操作结果是0
- Bit1 (DC): 四位进位与四位借位标记位
  - 加指令:
    - 0: 无低四位进位
    - 1: 从低四位进位
  - 减指令:
    - 0: 从低四位借位
    - 1: 无低四位借位
- Bit0 (C): 进位与借位标记位
  - 加指令:
    - 0: 无进位
    - 1: 从 MSB 进位
  - 减指令:
    - 0: 从MSB借位
    - 1: 无借位





## 7. 指令集

<注> 指令周末是系统时钟/2

JUMP INSTRUCTION				
LCALL I	Call subroutine. However, LCALL can addressing 16K address	2	None	011i iiiiii iiiiii
LGOTO I	Go branch to any address	2	None	010i iiiiii iiiiii
LOGIC				
AND M, a	$(M) \cdot (acc) \rightarrow (acc)$	1	Z	1010 1000 MMMM MMMM
AND M, m	$(M) \cdot (acc) \rightarrow (M)$	1	Z	1010 1001 MMMM MMMM
ANDLA I	Immediate $\cdot (acc) \rightarrow (acc)$	1	Z	1111 1000 iiiiii iiiiii
COM M, a	$\sim(M) \rightarrow (acc)$	1	Z	1010 0100 MMMM MMMM
COM M, m	$\sim(M) \rightarrow (M)$	1	Z	1010 0101 MMMM MMMM
IOR M, a	$(M) \text{ or } (acc) \rightarrow (acc)$	1	Z	1011 1110 MMMM MMMM
IOR M, m	$(M) \text{ or } (acc) \rightarrow (M)$	1	Z	1011 1111 MMMM MMMM
IORLA I	Immediate or $(acc) \rightarrow (acc)$	1	Z	1111 0010 iiiiii iiiiii
RL M, a	Rotate left from m to acc $m[6:0] \rightarrow acc[7:1]$ $m[7] \rightarrow acc[0]$	1	None	1110 0000 MMMM MMMM
RL M, m	Rotate left from m to itself $m[6:0] \rightarrow m[7:1]$ $m[7] \rightarrow m[0]$	1	None	1110 0001 MMMM MMMM
RLC M, a	Rotate left from m to acc $m[7] \rightarrow c$ $m[6:0] \rightarrow acc[7:1]$ $c \rightarrow acc[0]$	1	C	1110 0010 MMMM MMMM
RLC M, m	Rotate left from m to itself $m[7] \rightarrow c$ $m[6:0] \rightarrow m[7:1]$ $c \rightarrow m[0]$	1	C	1110 0011 MMMM MMMM
RR M, a	Rotate right from m to acc $0 \rightarrow acc[7]$ $m[7:1] \rightarrow acc[6:0]$	1	None	1110 1000 MMMM MMMM
RR M, m	Rotate right from m to itself $M[0] \rightarrow m[7]$	1	None	1110 1001 MMMM MMMM



	$m[7:1] \rightarrow m[6:0]$			
RRC M, a	Rotate right from m to acc $m[0] \rightarrow c, c \rightarrow acc[7]$ $m[7:1] \rightarrow acc[6:0]$	1	C	1110 1010 MMMM MMMM
RRC M, m	Rotate right from m to itself $m[0] \rightarrow c, c \rightarrow m[7]$ $m[7:1] \rightarrow m[6:0]$	1	C	1110 1011 MMMM MMMM
SWAP M, a	$m[7:4] \leftrightarrow acc[3:0]$ $m[3:0] \leftrightarrow acc[7:4]$	1	None	1011 1100 MMMM MMMM
SWAP M, m	$m[7:4] \leftrightarrow m[3:0]$	1	None	1011 1101 MMMM MMMM
XOR M, a	$(M) \text{ xor } (acc) \rightarrow (acc)$	1	Z	1011 0110 MMMM MMMM
XOR M, m	$(M) \text{ xor } (acc) \rightarrow (M)$	1	Z	1011 0111 MMMM MMMM
XORLA I	Immediate xor $(acc) \rightarrow (acc)$	1	Z	1111 1001 iiiii iiiii
<b>MATHEMATICS</b>				
ADD M, a	$(M) + (acc) \rightarrow (acc)$	1	C, DC, Z	1010 1010 MMMM MMMM
ADD M, m	$(M) + (acc) \rightarrow (M)$	1	C, DC, Z	1010 1011 MMMM MMMM
ADDC M,a	$(M) + (acc) + (\text{carry}) \rightarrow (acc)$	1	C, DC, Z	1011 1010 MMMM MMMM
ADDC M,m	$(M) + (acc) + (\text{carry}) \rightarrow (M)$	1	C, DC, Z	1011 1011 MMMM MMMM
ADDLA I	Immediate + $(acc) \rightarrow (acc)$	1	C, DC, Z	1111 1010 MMMM MMMM
BC M, bn	Clear bit n of (M)	1	None	1001 1bbb MMMM MMMM
BS M, bn	Set bit n of (M)	1	None	1001 0bbb MMMM MMMM
CLRA	Clear accumulator	1	Z	1010 0010 0000 0000
CLR M	Clear memory M	1	Z	1010 0011 MMMM MMMM
TABRDL M	Read low byte ROM table to (acc) ROM table address={TB_BNK,index of M }	2	None	1101 1000 MMMM MMMM
TABRDH M	Read high byte ROM table to (acc) ROM table address={TB_BNK,index of M }	2	None	1101 1001 MMMM MMMM
SUBC M, a	$(M) + (/acc) + (\text{carry}) \rightarrow (acc)$	1	C, DC, Z	1101 0100 MMMM MMMM
SUBC M, m	$(M) + (/acc) + (\text{carry}) \rightarrow (M)$	1	C, DC, Z	1101 0101 MMMM MMMM
DA M, a	Decimal Adjust M to ACC If $ACC[3:0] > 9$ or $DC=1$ Then $ACC[3:0] \leftarrow ACC[3:0] + 6,$ $DC1 \leftarrow \neg DC$ else $ACC[3:0] \leftarrow ACC[3:0],$	1	C	1101 0110 MMMM MMMM



	DC1=0 If ACC[7:4]+DC1 > 9 or C=1 Then ACC[7:4] $\mathbf{\bar{B}}$ ACC[7:4]+6+DC1, C=1 else ACC[7:4] $\mathbf{B}$ ACC[7:4]+DC1, C=C			
DA M, m	Decimal Adjust M to memory If ACC[3:0] > 9 or DC=1 Then M[3:0] $\mathbf{\bar{B}}$ ACC[3:0]+6, DC1=-DC else M[3:0] $\mathbf{B}$ ACC[3:0], DC1=0 If ACC[7:4]+DC1 > 9 or C=1 Then M[7:4] $\mathbf{\bar{B}}$ ACC[7:4]+6+DC1, C=1 else M[7:4] $\mathbf{B}$ ACC[7:4]+DC1, C=C	1	C	1101 0111 MMMM MMMM
DEC M, a	(M) - 1 $\rightarrow$ (acc)	1	Z	1010 1100 MMMM MMMM
DEC M, m	(M) - 1 $\rightarrow$ (M)	1	Z	1010 1101 MMMM MMMM
INC M, a	(M) + 1 $\rightarrow$ (acc)	1	Z	1011 0000 MMMM MMMM
INC M, m	(M) + 1 $\rightarrow$ (M)	1	Z	1011 0001 MMMM MMMM
MOVAM m	(acc) $\mathbf{\bar{a}}$ (M)	1	None	1010 0001 MMMM MMMM
MOV M, a	(M) $\rightarrow$ (acc)	1	Z	1010 0110 MMMM MMMM
MOV M, m	(M) $\rightarrow$ (M)	1	Z	1010 0111 MMMM MMMM
MOVLA I	Immediate data $\mathbf{\bar{a}}$ acc	1	None	1111 0000 iiiii iiiii
SUBLA I	(immediate data)-(Acc) $\rightarrow$ (Acc)	1	C, DC, Z	1111 0100 iiiii iiiii
SUB M, m	(M)-(acc) $\rightarrow$ (M)	1	C, DC, Z	1011 0101 MMMM MMMM
SUB M, a	(M)-(acc) $\rightarrow$ (acc)	1	C, DC, Z	1011 0100 MMMM MMMM
<b>OTHER OPERATION</b>				
NOP	No operation	1	None	1111 1111 1111 1111
CLRWDT	Clear watch-dog register	1	$\overline{TO}, \overline{PD}$	1111 1111 1111 0000
RET	Return (for lcall instruction )	2	None	1111 1111 1111 0001
RETI	Return and enable INTM( for IRQ )	2	None	1111 1111 1111 0010
RET_INT	Return ( for IRQ )	2	None	1111 1111 1111 0011
SLEEP	Enter sleep (saving) mode	1	$\overline{TO}, \overline{PD}$	1111 1111 1111 0100
<b>CONDITION OPERATION</b>				
BTSC M, bn	If (bit n of (M))=0, skip next instruction	1 or 2	None	1000 1bbb MMMM MMMM
BTSS M, bn	If (bit n of (M))=1, skip next instruction	1 or 2	None	1000 0bbb MMMM MMMM
DECSZ M, a	(M) - 1 $\rightarrow$ (acc), skip if (acc) = 0	1 or 2	None	1010 1110 MMMM MMMM



DECSZ M, m	(M) - 1 → (M), skip if (M) = 0	1 or 2	None	1010 1111 MMMM MMMM
INCSZ M, a	(M) + 1 →(acc), skip if (acc) = 0	1 or 2	None	1011 0010 MMMM MMMM
INCSZ M, m	(M) + 1 → (M), skip if (M) = 0	1 or 2	None	1011 0011 MMMM MMMM
TMSS A	If (acc) =0, skip next instruction	1 or 2	None	1011 1000 XXXX XXXX
TMSC M	If (M) = 0, skip next instruction	1 or 2	None	1011 1001 MMMM MMMM

< 注 > 进入 SLEEP 指令后，请增加一个 NOP 指令在它后面来进行延时。



## 8. 电气特性

### 8.1 绝对最大额定值

电源电压 .....	$V_{SS}-0.3V$ to $V_{SS}+5.5V$	存储温度 .....	$-50^{\circ}C$ to $125^{\circ}C$
输入电压 .....	$V_{SS}-0.3V$ to $V_{DD}+0.3V$	工作温度 .....	$-40^{\circ}C$ to $85^{\circ}C$

< 注 > 这里仅强调额定值，超出“绝对最大额定值”指定的范围会对芯片造成严重伤害。在其他条件下（规格书列出的除外），此芯片的功能操作并不意味长期暴露在极端条件下会影响芯片的可靠性。



## 8.2 直流电特性

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		VDD	Conditions				
VDD	Operating Voltage	---	f <sub>sys</sub> =4MHZ	2.0		5.5	V
			f <sub>sys</sub> =8MHZ	2.5		5.5	
			f <sub>sys</sub> =12MHZ	3.3		5.5	
I <sub>DD1</sub>	Operating Current (Crystal OSC)	3.3V	No Load , f <sub>sys</sub> =4MHZ ADC disable .		0.8		mA
		5V			1.6		
I <sub>DD2</sub>	Operating Current (RC OSC)	3.3V	No Load , f <sub>sys</sub> =4MHZ ADC disable .		0.7		mA
		5V			1.2		
I <sub>DD3</sub>	Operating Current (Crystal OSC)	5V	No Load , f <sub>sys</sub> =8MHZ ADC disable		2.6		mA
I <sub>DD4</sub>	Operating Current (Crystal OSC)	5V	No Load , f <sub>sys</sub> =12MHZ ADC disable		3.6		mA
V <sub>IH</sub>	Input High Voltage	5V	I/O Port	2		VDD	V
V <sub>IL</sub>	Input Low Voltage	5V	I/O Port			0.8	V
I <sub>STB</sub>	Standby Current	5V	WDT disable			1	μA
			WDT enable			6	
		3V	WDT disable			1	
			WDT enable			2	
I <sub>IL</sub>	Input Leakage Current	5V	Vin=VDD, VSS			1	μA
I <sub>OH</sub>	I/O Port Driving Current	5.5V	Voh=5V			9.9	mA
			Voh=4.5V			17.6	
			Voh=4V			24.8	
I <sub>OL</sub>	I/O Port Sink Current	5.5V	Vol=0.5V			24.5	mA
			Vol=0.75V			35.3	
			Vol=1V			43.8	
R <sub>PH</sub>	Pull-high Resistance	5V		70	85	100	KΩ
		3.3V		120	150	180	KΩ
V <sub>AD</sub>	A/D input Voltage			0		VDD	V
E <sub>AD</sub>	A/D Conversion Error					2	LSB
I <sub>ADC</sub>	A/D Operating current	3.3V			100		uA

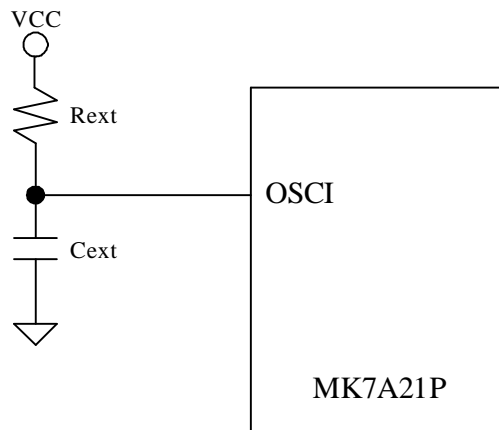


### 8.3 交流电特性

Symbol	Parameter	Test Conditions		Min	Typ.	Max	Unit
		Conditions	VDD				
f <sub>sys1</sub>	System Clock	LS Crystal mode	5V	32		200	Khz
			3V	32		200	
f <sub>sys2</sub>	System Clock	NS Crystal mode	5V	0.2		10	Mhz
			3V	0.2		10	
f <sub>sys3</sub>	System Clock	HS Crystal mode	5V	10		20	Mhz
f <sub>sys4</sub>	System Clock	RC mode	5V	3.6	4	4.4	Mhz
			3V	3.6	4	4.4	
T <sub>wdt</sub>	Watchdog Timer		5V		20		mS
			3V				
T <sub>rht</sub>	Reset Hold Time		5V		20		mS
			3V				
T <sub>AD</sub>	A/D clock period			3			us
T <sub>ADC</sub>	A/D Conversion Time				30		t <sub>AD</sub>
T <sub>ADCS</sub>	A/D Sampling Time				8		t <sub>AD</sub>
I <sub>OH</sub>	Output high driving current	5V	Voh=0.9VDD		-10		mA
		3V	Voh=0.9VDD		-4		mA
I <sub>OL</sub>	output low sink current	5V	Voh=0.1VDD		20		mA
		3V	Voh=0.1VDD		8		mA



### 8.4 EXT\_RC 振荡器频率



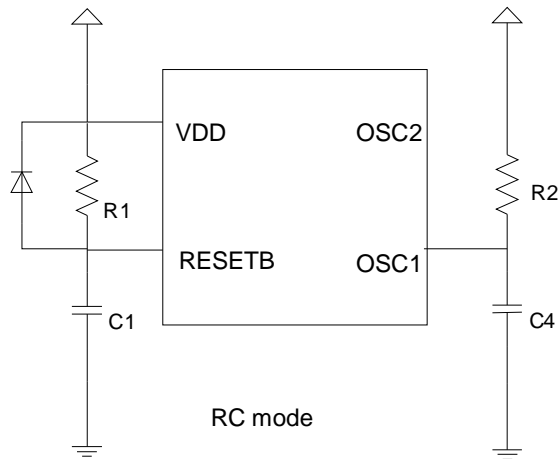
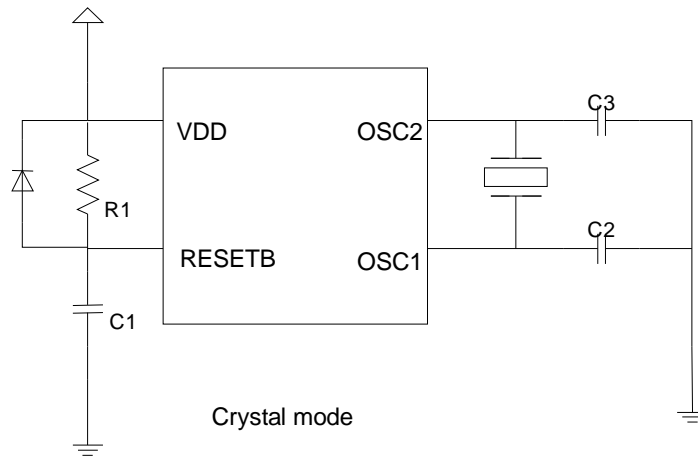
下表为典型的外部 RC 振荡频率数据表  
当  $C_{ext} = 0.1\mu f$  (104)

Rext	5V	3V
6M	32KHZ	30KHZ
300K	460 KHZ	450 KHZ
140K	0.97 MHZ	0.90 MHZ
68K	2.0 MHZ	1.97 MHZ
37K	3.9 MHZ	3.85 MHZ
25K	6 MHZ	6 MHZ
15K	10.3 MHZ	10 MHZ





8.5 不同模式下的振荡器电路

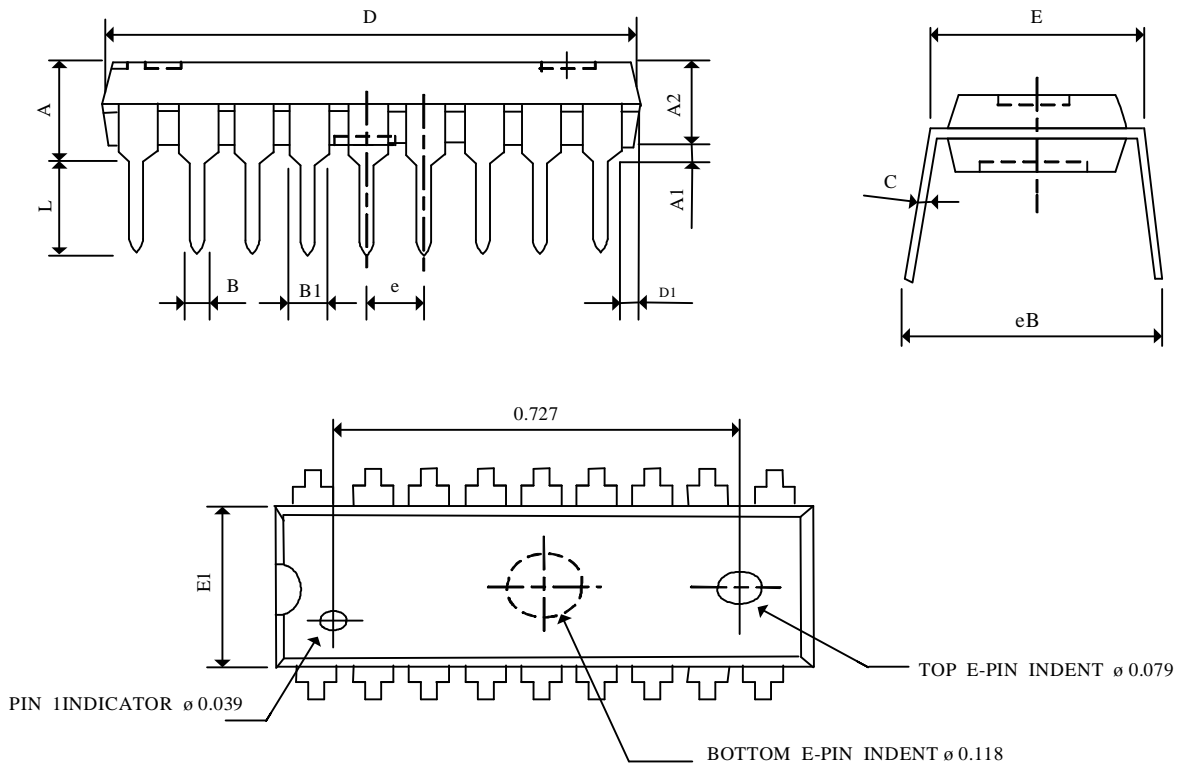


Crystal mode	Crystal	C2	C3
LP mode	32 KHz	20~100P	20~100P
	455 KHz	100P	100P
NT mode	455 KHz	100P	100P
	1 Mhz	20~100P	20~100P
	2 Mhz	20~100P	20~100P
	4 Mhz	20~100P	20~100P
	8 Mhz	20~50P	20~50P
	10 Mhz	20~50P	20~50P
HS mode	12 Mhz	20~50P	20~50P
	16 Mhz	20~50P	20~50P
	20 Mhz	20~50P	20~50P



## 9. 封装尺寸

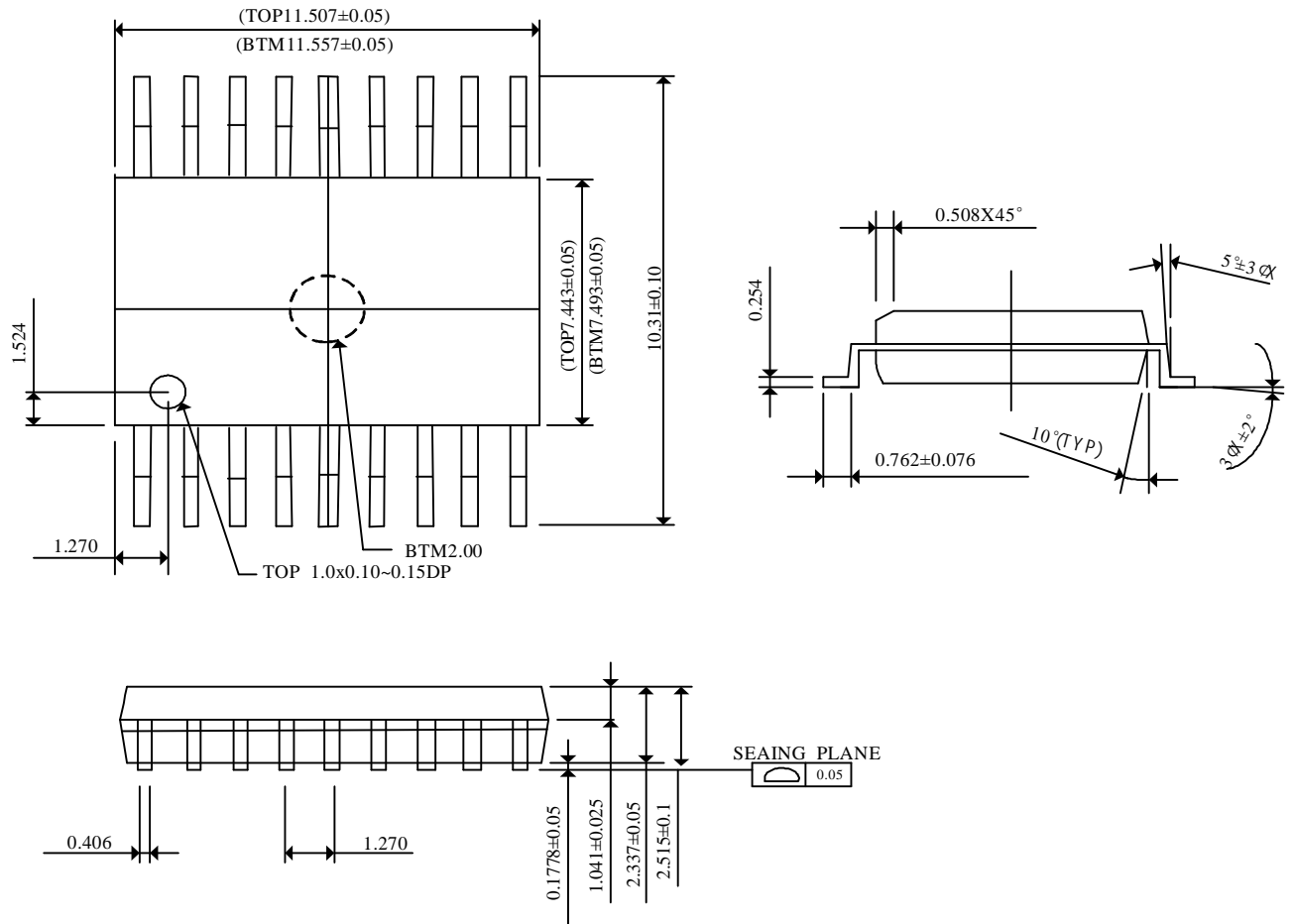
(a) 18 Pin DIP



SYMBOLS	DIMENSIONS IN MILLIMETERS			DIMENSIONS IN INCHES		
	MIN	NOM	MAX	MIN	NOM	MAX
A	—	—	4.57	—	—	0.180
A1	0.38	—	—	0.015	—	—
A2	—	3.30	3.56	—	0.130	0.140
B	0.36	0.46	0.56	0.014	0.018	0.022
B1	1.27	1.52	1.78	0.050	0.060	0.070
C	0.20	0.25	0.33	0.008	0.010	0.013
D	22.71	22.96	23.11	0.894	0.904	0.910
D1	0.43	0.56	0.69	0.017	0.022	0.027
E	7.62	—	8.26	0.300	—	0.325
E1	6.40	6.50	6.65	0.252	0.256	0.262
e	—	2.54	—	—	0.100	—
L	3.18	—	—	0.125	—	—
eB	8.38	—	9.65	0.330	—	0.380

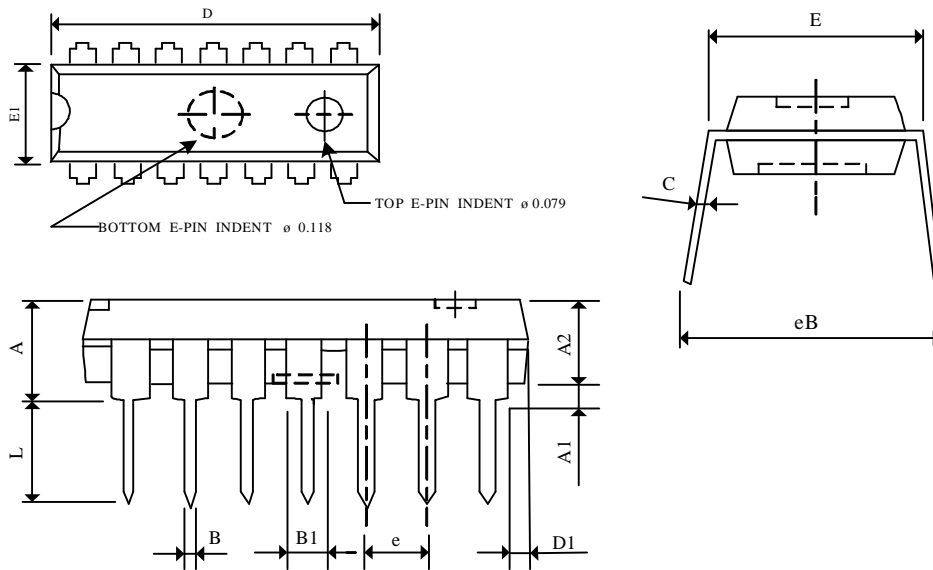


(b) 18 Pin SOP





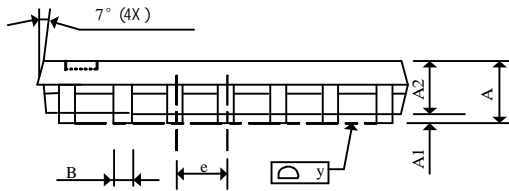
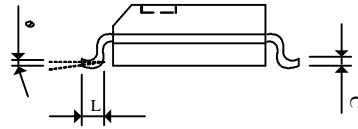
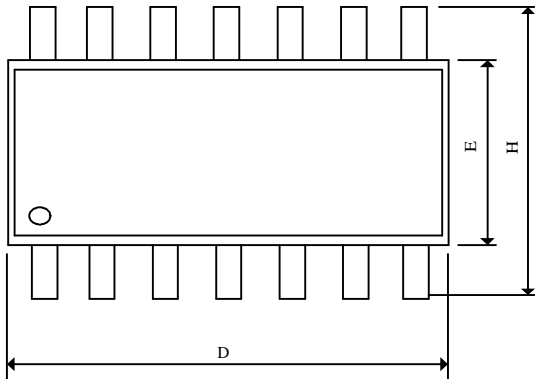
(c) 14 Pin DIP



SYMBOLS	DIMENSIONS IN MILLIMETERS			DIMENSIONS IN INCHES		
	MIN	NOM	MAX	MIN	NOM	MAX
A	—	—	4.57	—	—	0.180
A1	0.38	—	—	0.015	—	—
A2	3.25	3.30	3.45	0.128	0.130	0.136
B	0.36	0.46	0.56	0.014	0.018	0.022
B1	1.27	1.52	1.78	0.050	0.060	0.070
C	0.20	0.25	0.33	0.008	0.010	0.013
D	18.90	19.15	19.30	0.744	0.754	0.760
D1	1.07	1.19	1.32	0.042	0.047	0.052
E	7.62	—	8.26	0.300	—	0.325
E1	6.35	6.50	6.65	0.250	0.256	0.262
e	—	2.54	—	—	0.100	—
L	3.18	—	—	0.125	—	—
eB	8.64	—	9.65	0.340	—	0.380



(d) 14 Pin SOP



SYMBOLS	DIMENSIONS IN MILLIMETERS			DIMENSIONS IN INCHES		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.60	1.75	0.053	0.063	0.069
A1	0.10	—	0.25	0.004	—	0.010
A2	—	1.45	—	—	0.057	—
B	0.33	—	0.51	0.013	—	0.020
C	0.19	—	0.25	0.007	—	0.010
D	8.55	—	8.75	0.337	—	0.344
E	3.80	—	4.00	0.150	—	0.157
e	—	1.27	—	—	0.050	—
H	5.80	—	6.20	0.228	—	0.244
L	0.40	—	1.27	0.016	—	0.050
Y	—	—	0.10	—	—	0.004
θ	0°	—	8°	0°	—	8°



**<郑重声明>**

Mikkon 产品未设计授权使用作为以下系统设备元件：外科移植，原子能控制仪器，飞机或太空船仪器，交通运输设备，交通信号仪器，燃烧控制设备，或其他应用于为了维持生命的场合。此外，Mikkon 产品也不适用于由于 Mikkon 产品失效可能引起或导致个人受伤，死亡，严重的财产及环境破坏的情况。

Mikkon 的客户在使用或销售 Mikkon 产品做上述用途时，必须自行承担风险，同时承诺完全赔偿 Mikkon 因不当使用或销售带来的任何损失。