

(Version 1.0: 2020-01-03)

目录

(Version 1.0: 2020-01-03)	1
一、简介	2
二、特征	2
三、管脚分配	4
封装脚位图	4
管脚描述	5
PAD 图	8
衬底必须接至 VSS	8
四、存储器	9
4.1 程序存储器	9
4.2 数据存储器	10
4.3 特殊功能寄存器	10
4.4 特殊功能寄存器表	10
4.5 特殊功能寄存器描述	12
五、功能描述	43
5.1 TCC/WDT 预分频器	43
5.2 I/O 端口	44
5.3 定时器	45
• 概述	45
• 功能描述	45
• 可编程相关寄存器组	46
5.4 复位和唤醒	46
5.5 中断	48
中断输入电路	48
5.6 振荡器	48
六、绝对最大范围	49
七、电气特性	49
八、封装尺寸图	51
SOP24	51
SOP14	52
注意:	52
附: 版本记录	52

一、简介

TX322是一个具有低功耗和高速的8位微处理器。它的操作核心由RISC类体系结构实现。有一个脉冲宽度调制输出。同时也增强了单片机的其它内部特性，如暂停、唤醒功能、振荡器选择和可编程分频器等，增加了单片机的使用灵活度，而这些特性也同时保证实际应用时只需要最少的外部器件，进而降低了整个产品的成本。此单片机广泛被应用在红外遥控发射器、马达驱动、工业控制、消费性产品和子系统控制器等场合。

采用 16 位精简指令集并拥有高达 256 字节的 RAM，程序可方便的访问控制器内部的 ROM 空间及 RAM 空间而不需要执行繁琐的换页功能。采用特殊功能寄存器与 RAM 统一寻址的方式。

TX322是属于一次可编程(One-Time Programmable, OTP)单片机，配合使用我们提供的程序开发工具，可简单有效的更新程序，给设计者提供了快速有效的开发途径。

二、特征

- 高性能、低功耗的8位微处理器
- 先进的RISC结构
 - 59条指令 – 大多数指令执行时间为单个指令周期（振荡频率/2）
- 外设特点
 - 一个具有预分频器及中断功能的 8 位定时器
 - 一个具有预分频器及中断功能的 8 位脉宽比较器
 - 一个具有独立振荡器的内部看门狗
 - 一个与 IO 口复用的外部中断输入
 - 芯片内置晶体及电阻电容振荡器
 - 烧录电路接口及程序代码保护功能
 - 低电压复位（LVR）特性
 - P5, P6, P7 口具有唤醒功能
 - 3 通道 8 位的 PWM
 - 1 个 UART 接口
 - 新增 Green mode, 进入 Green mode, 唤醒可由
 1. 外部中断 (/INT) ;
 2. TCC 溢出中断;
 3. Timer1 比较器匹配中断;此外, PWM1 / PWM2/ PWM3 于 Green-mode 仍然可以输出.
- 特殊的处理器特点
 - 上电复位及掉电检测
 - 片内经过标定的 RC 振荡器

- 片内/片外有 3 个硬件中断源
 1. 外部中断 (/INT)。
 2. TCC 溢出中断。
 3. Timer1 比较器匹配中断。
 4. IO 口改变中断 (从睡眠模式唤醒)
 5. WDT timeout 中断。
 6. Low-voltage detection 中断。
 7. PTMR1/PWM1, PTMR3/PWM3 比较器匹配中断。
 8. PTMR2/PWM2 比较器匹配中断。
 9. UART 中断。

- **16 级堆栈 (CALL or PUSH)，使用 32 * 8bit RAM (192 * 8bit 中的高 32 位)**
- 两种睡眠模式，暂停与唤醒特性可以节省功耗
- 指令为 16 位宽，对所有寄存器、RAM 的访问都不需要分页；程序 ROM 也不需要分页
- 表格读取功能
- 直接和间接数据寻址模式
- 位操作指令

- 存储器
 - 4K * 16bit 程序存储器 (OTP ROM)
 - 192 * 8bit 数据存储器 (user 空间 160x8 + stack 32x8)
 - 128 * 8bit 特殊功能寄存器

- IO 和封装
 - 22 个可用 IO

- 适用的温度范围： -40 ~ 85° C

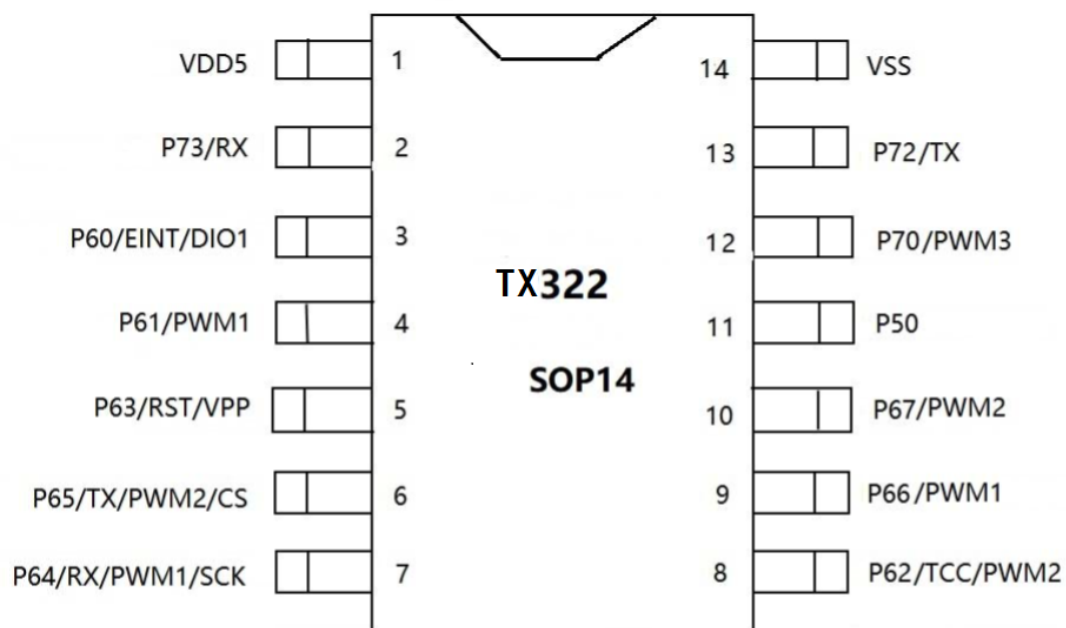
- 工作电压范围： 1.8 ~ 5.5V

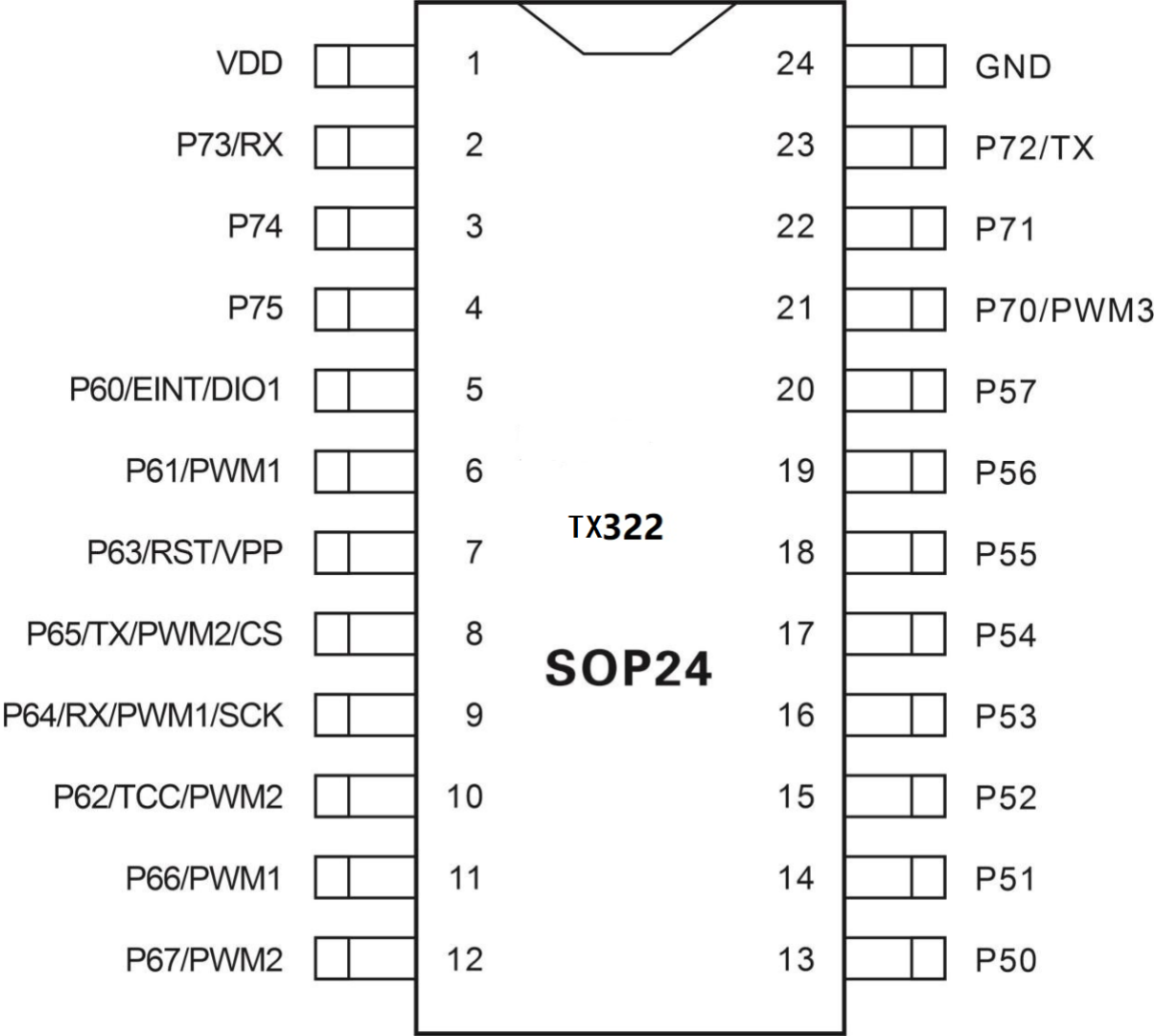
- 工作频率范围
 - i. RC 类型： 有 8MHz、4MHz、1MHz、455KHz 可以选。

- TX322 25° C 时的功耗：
 1. 正常模式：在 5V/4MHz 时，小于 1.5mA
 2. 睡眠模式：小于 2uA

三、管脚分配

封装脚位图





管脚描述

引脚名称	引脚号	I/O/P类型	缓冲器类型	功能
P50		I/O	TTL	数字 I/O 引脚，可软件设为下拉
P51		I/O	TTL	数字 I/O 引脚，可软件设为下拉

TX322 8-BIT MCU

P52		I/O	TTL	数字 I/O 引脚, 可软件设为下拉
P53		I/O	TTL	数字 I/O 引脚, 可软件设为下拉
P54		I/O	TTL	数字 I/O 引脚, 可软件设为下拉
P55		I/O	TTL	数字 I/O 引脚, 可软件设为下拉
P56		I/O	TTL	数字 I/O 引脚, 可软件设为下拉
P57		I/O	TTL	数字 I/O 引脚, 可软件设为下拉
P60/EINT P60 EINT *TDIO1		I/O I I/O	ST/TTL ST ST	数字 I/O 引脚,集电极开路输出与内部上下拉,IO 口改变从睡眠模式唤醒 外部中断脚,下降沿触发 Testing interface DIO1 (烧录数据脚)
P61 / PWM1 P61 PWM1		I/O O	ST/TTL	数字 I/O 引脚,集电极开路输出与内部上下拉,IO 口改变从睡眠模式唤醒 PWM1 输出
P62/TCC / PWM2 P62 TCC PWM2		I/O I O	ST/TTL ST	数字 I/O 引脚,集电极开路输出与内部上下拉,IO 口改变从睡眠模式唤醒 外部时钟与计数输入脚 PWM2 输出
P63/RST P63 RST *VPP		I/O I	ST/TTL ST	数字 I/O 引脚,集电极开路输出与内部上下拉, IO 口改变从睡眠模式唤醒 当选择复位脚时为复位功能,低电平复位 OTP 烧录时,此脚是烧录脚 VPP, 7.3V 高压
P64/RX / PWM1 P64 RX PWM1 *TSCK		I/O I O I	ST/TTL ST ST	数字 I/O 引脚,集电极开路输出与内部上拉,IO 口改变从睡眠模式唤醒 UART RX PWM1 输出 Testing interface SCK (烧录 CLK 脚)
P65/TX / PWM2 P65 TX PWM2 *TCS		I/O O O I	ST/TTL ST	数字 I/O 引脚,集电极开路输出与内部上拉,IO 口改变从睡眠模式唤醒 UART TX PWM2 输出 Testing interface CS (烧录 CS 脚)
P66 / PWM1 P66 PWM1		I/O O	ST/TTL	数字 I/O 引脚,集电极开路输出与内部上拉,IO 口改变从睡眠模式唤醒 PWM1
P67 /PWM2 P67 PWM2		I/O O	ST/TTL	数字 I/O 引脚,集电极开路输出与内部上拉,IO 口改变从睡眠模式唤醒 PWM2
P70 /PWM3 P70 PWM3		I/O O	ST/TTL	数字 I/O 引脚,内部上拉,IO 口改变从睡眠模式唤醒 PWM3
P71		I/O	TTL	数字 I/O 引脚, 可软件设为上拉,IO 口改变从睡眠模式唤醒

TX322 8-BIT MCU

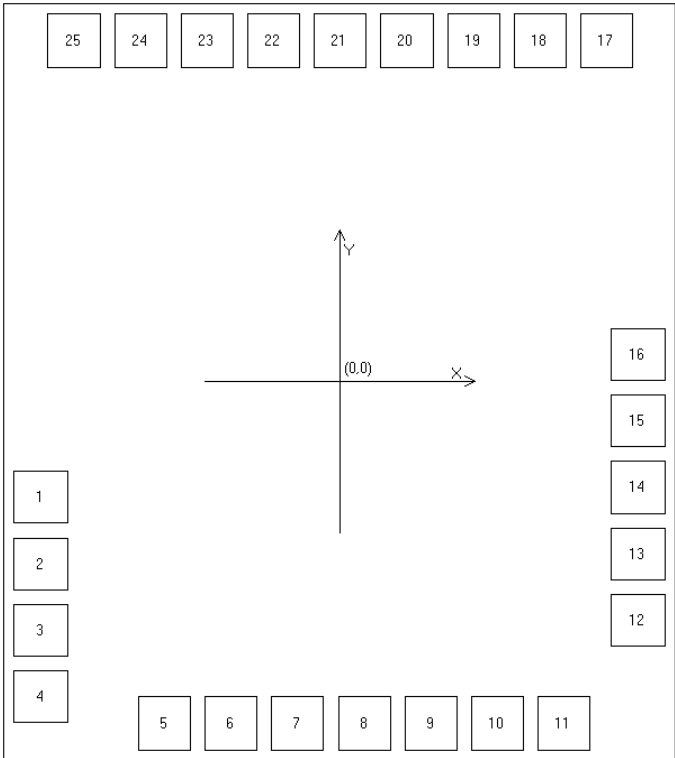
P72/TX P72 TX		I/O O	ST/TTL	数字 I/O 引脚,集电极开路输出与内部上拉,IO 口改变从睡眠模式唤醒 UART TX
P73/RX P73 RX		I/O I	ST/TTL ST	数字 I/O 引脚,内部上拉,IO 口改变从睡眠模式唤醒 UART RX
P74		I/O	TTL	数字 I/O 引脚, 可软件设为上拉,IO 口改变从睡眠模式唤醒
P75		I/O	TTL	数字 I/O 引脚, 可软件设为上拉,IO 口改变从睡眠模式唤醒
VDD		P		电源输入, 烧录脚
VDDL		P		IC 的内核电压输出, 烧录脚, 连接 0.1uF 的电容到 GND
VSS		P		接地端, 烧录脚

图注: I = 输入 O = 输出 I/O = 输入/输出 P = 电源

- = 未使用 TTL = TTL 输入 ST = 施密特输入

注:1.实际管脚顺序以下面 PAD 图为准

PAD 图



衬底必须接至 VSS

	PAD NAME		NO.	PAD NAME
1	p6[3]		14	p5[7]
2	p6[5]		15	p7[0]
3	p6[4]		16	p7[1]
4	p6[2]		17	p7[2]
5	p6[6]		18	VSS
6	p6[7]		19	VDD
7	p5[0]		20	VDD5
8	p5[1]		21	p7[3]
9	p5[2]		22	p7[4]
10	p5[3]		23	p7[5]
11	p5[4]		24	p6[0]
12	p5[5]		25	p6[1]
13	p5[6]			

四、存储器

本节讲述 TX322 的存储器。TX322 具有两个主要的存储器空间：数据存储器空间和程序存储器空间。此外，TX322 还有寄存器区对 IC 外设进行特定的操作。数据存储器空间及寄存器空间统一编址，并与程序存储器空间分开。

4.1 程序存储器

4K * 16bit 的 OTP ROM 空间，复位矢量地址为 0X000，硬件中断矢量地址为 0X003~0X01F。

复位入口地址	0X000
Low-voltage detection 中断入口地址	0X003
外部 INT 中断入口地址	0X005
IO 口改变唤醒中断入口地址	0X007
定时器 TCC 中断入口地址	0X009
看门狗 WDT 溢出中断入口地址	0X00B
PWP (TIMER1) 中断入口地址	0X011
PTMR1/PWM1、PTMR3/PWM3 中断入口地址	0X013
PTMR2/PWM2 中断入口地址	0X015
UART 中断入口地址	0X01D
程序存储器空间	0X020
	0XFFF

4.2 数据存储器

256 * 8bit 的 RAM 空间， 有两种寻址方式：直接寻址及通过 INDF 间接寻址。

特殊功能寄存器空间	0X17F
	0X100
保留空间	0X0FF
	0X0C0
堆栈空间	0X0BF
	0X0A0
	0X09F
数据存储器空间	0X000

4.3 特殊功能寄存器

128 * 8bit 的特殊功能寄存器空间采用非连续编址的形式。对于没有对应寄存器的地址进行读写操作，结果未定义。

特殊功能寄存器只能采用直接寻址方式。

4.4 特殊功能寄存器表

TX322 8-BIT MCU

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INDF	100h	用 FSR 的内容寻址数据存储器来寻址该地址单元（非实际存在的寄存器）							
TCC	101h	TCC 模块的寄存器							
PCL	102h	程序计数器（PC）的低字节							
STATUS	103h	RST	GP1	GP0	T	P	Z	DC	C
FSR	104h	间接数据存储器地址指针							
P5D	105h	P5 数据锁存器							
P6D	106h	P6 数据锁存器							
P7D	107h	P7 数据锁存器							
P5S	115h	读取 P5 的引脚电平							
P6S	116h	读取 P6 的引脚电平							
P7S	117h	读取 P7 的引脚电平							
WUCON6	120h	WU67	WU66	WU65	WU64	WU63	WU62	WU61	WU60
WUCON5	121h	WU57	WU56	WU55	WU54	WU53	WU52	WU51	WU50
CONT(R/W)	122h	TS1	/INT	TS0	TE	PAB	PSR2	PSR1	PSR0
OPEN_DRAIN	123h	OD67	OD66	OD65	OD64	OD63	OD62	OD61	OD60
IOC5	125h	P5 数据方向寄存器							
IOC6	126h	P6 数据方向寄存器							
IOC7	127h	P7 数据方向寄存器							
PULL_P5H	12Ah	/PH57	/PH56	/PH55	/PH54	/PD57	/PD56	/PD55	/PD54
PULL_DOWN	12Bh	/PD63	/PD62	/PD61	/PD60	/PD53	/PD52	/PD51	/PD50
TICON	12Ch						TM1E	TM1P1	TM1P0
PULL_HIGH	12Dh	/PH67	/PH66	/PH65	/PH64	/PH63	/PH62	/PH61	/PH60
WDTCON	12Eh	EIS	EI_ES	WDTE	SLPC	WDT_CKS			WU7
INTIE	12Fh	WDT_IE			ICIE	TM1IE		EXIE	TCIE
PWM1_DC	130h	PWM1 占空比寄存器							
PWM2_DC	131h	PWM2 占空比寄存器							
PWM1_PR	132h	PWM1 周期寄存器							
PWM2_PR	133h	PWM1 周期寄存器							
PWM1CON	134h	PEN2	PEN1	PS2[2]	PS2[1]	PS2[0]	PS1[2]	PS1[1]	PS1[0]
PTMR1CON	135h	PTM1E	PTM1IE	PTM1IF	PTM2E	PTM2IE	PTM2IF	PO_INV	PTM12CA
TMR1	13Eh	Timer1 模块的寄存器							
PWP	13Fh	脉宽设置寄存器							
PAL	14Bh	PA[6:0]							PA_SEL
PAH	14Ch				PA[11]	PA[10]	PA[9]	PA[8]	PA[7]
UA_CON	150h	UA_EN	RX_EN	RX_IE	TX_IE	BAUD16	TXOF/TXF	RXF	RXOF
UA_BAUD	151h	UART Baud-rate register							
UA_BUF	152h	UART data register							
IO_PULL2	15Fh					/P7_PH			

TX322 8-BIT MCU

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IO_PULL2	15Fh					/P7_PH			
IO_SINK2	171h					P7_XS			P5H_XS
IO_FUNC	172h				F_CKO[1]	F_CKO[0]	F_UART	F_PWM[1]	F_PWM[0]
IO_BUF2	173h					P7_ST			P5H_ST
IO_SINK	174h	P523_XS	P501_XS	P67_XS	P66_XS	P645_XS	P623_XS	P61_XS	P60_XS
IO_BUF	175h	P523_ST	P501_ST	P67_ST	P66_ST	P645_ST	P623_ST	P61_ST	P60_ST
OSCCON	178h		GREEN						
GCKCON	17Ch	G_SYS	G_GIO	G_TMR1				G_UART	G_PWM
LVRCON	17Dh	LVR_ENB	LVD_F	LVD_IE	LVD2_ENB	FT_EN	LVS2	LVS1	LVS0
INTIF	17Fh	WDT_IF			ICIF	TM1IF		EXIF	TCIF
PWM3_DC	180h	PWM3 占空比寄存器							
PWM3_PR	182h	PWM3 周期寄存器							
PWM3CON	184h		PEN3				PS3[2]	PS3[1]	PS3[0]
PTMR3CON	185h	PTM3E	PTM3IE	PTM3IF				PO3_INV	

4.5 特殊功能寄存器描述

说明：

- R：可读，W：可写，C：可清零
- RESET：复位状态
- U：表示无变化
- x：表示不确定

A(累加器)

内部数据传输，或者指令操作数保持。它是一个不可寻址寄存器。

CONT(控制寄存器) (地址:0x122)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	TS1	/INT	TS0	TE	PAB	PSR2	PSR1	PSR0
R/W/C	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	0	1	1	1	1	1	1
WDT_out	1	0	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

Bit6(/INT) 中断使能位，Read Only。

0：由指令 DISI 设置中断屏蔽；

1：由指令 ENI/RETI 设置中断使能。

Bit7-Bit5(TS1-TS0) TCC 时钟源的选择位。

11：内部指令周期做时钟源；

00：由外部 TCC 脚来做时钟源。

01：OSCO 周期做时钟源；(When CKSRC = RCOSC)

10: reserved.

Bit4(TE) 外部 TCC 触发源选择位

0: 外部 TCC 上升沿计数;

1: 外部 TCC 下降沿计数。

Bit3(PAB)预分频器分配位。

0: 预分频器分配给 TCC;

1: 预分频器分配给 WDT，当预分频器分配给 WDT 后，TCC 为 1:1 分频。

Bit0(PSR0) ~ Bit2(PSR20 TCC/WDT 预分频位

PSR2	PSR1	PSR0	TCC Rate	WDT Rate
0	0	0	1:2	1:1
0	0	1	1:4	1:2
0	1	0	1:8	1:4
0	1	1	1:16	1:8
1	0	0	1:32	1:16
1	0	1	1:64	1:32
1	1	0	1:128	1:64
1	1	1	1:256	1:128

INDF(间接寻址寄存器)(地址:0x100)

INDF 寄存器不是实际存在的寄存器。它用于间接地址的指针。任何对 INDF 进行操作的指令，实际上是存取由 RAM 选择寄存器 FSR 所指定的 RAM 内容。

TCC(定时寄存器) (地址:0x101)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	D7	D6	D5	D4	D3	D2	D1	D0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

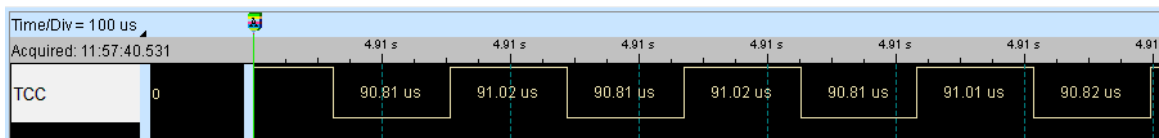
此寄存器为 8 位定时器，可由程序进行读/写操作。它用于对内部时钟计数，并拥有可编程的预分器（最大 256 分频）。

- 以下例子为 TCC 从 0x20 开始，数到 0xFF 后进中断改变 P50 状态(1、0 交替)，中断的频率为 11.1khz。

```

3 ORG 0x0000
4 JMP _START
5 ORG 0x0009 ;TCC中断入口地址
6 JMP _INT_TCC
7
8 _INT_TCC:
9  MOV A, @0x20
10 MOV TCC, A ;TCC重设 从0x20开始，若无重设会从0开始
11 MOV A, @0x01
12 XOR P5D, A
13 CLR INTIF ;TCC中断标志清除
14 RETI
15
16 ORG 0x0100
17 _START:
18  CALL _INIT_IO ;设置I/O管脚，P5 output low
19
20  MOV A, @0x20
21  MOV TCC, A ;TCC从0x20开始
22
23  MOV A, @0xA0
24  MOV CONT, A ;内部指令周期做时钟源，预分频器分配给TCC,TCC Rate=1:2;
25
26  BS INTIE, 0 ;TCC中断允许
27  ENI ;中断开启
28
29  NOP
30  JMP $-1;

```



$$F_{osc} = 10M, F_{sys} = 10M/2 = 5M$$

$$\text{预分频} = 1:2$$

$$0x20 = 32, 256-32 = 224$$

$$\text{TCC 中断时间} = 224((1/5M) * 2) = 89.6\mu s$$

PCL(程序计数器)和堆栈(地址:0x102)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

PC 和硬件堆栈为 16 位宽。

产生 4096×16 位的片内 OTP ROM 地址获取指令代码。

复位后 PC 所有位均设置为 ‘1’。

- “ADD R2, A”允许相对地址被装入当前 PC，同时 PC 的高 8 位被清零。
- 任何有可能改变 PCL 的值的指令(例如：“ADD PCL, A”，“MOV PCL, A”，“BC PCL, 6”，...)(“TBL”除外)都将导致 PC 的高 8 位被清零。因此，跳转限制在编程页的前 256 个程序存储空间。
- “TBL”允许相对地址被加载到当前 PC(PCL + A → PCL)，PC 的高 8 位的值不变。因此，跳转可以在编程页的任意 256 个程序存储空间。
- 除了改变 PCL 内容指令需要一个以上指令周期外(fclk/2)，其余指令只要一个指令周期。

STATUS(状态寄存器) (地址:0x103)

状态寄存器包含 ALU 算术结果的状态和复位状态。

与其他寄存器相同，状态寄存器可以作为任何指令的目的操作数。如果状态寄存器作为一条指令的目的操作数，而这条指令又影响了 Z、DC 或 C 标志，那么就不允许对这 3 位进行写操作。这些位的位置或清零取决于器件的逻辑模块。此外，T 和 P 位是不能进行写操作的，所以当执行一条把状态寄存器作为目的操作数的指令时，其结果可能会与预想的不同。

因此，如果想改变状态寄存器的内容，建议使用位操作指令 BC、BS 半字节交换指令 SWAP 和传送指令 MOV R, A，因为这些指令不影响 Z、C 或 DC 标志位。

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	RST	GP1	GP0	T	P	Z	DC	C
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	1	1	0	0	0
WDT_out	0	0	0	0	U	0	0	0
Wake_up	U	U	U	U	U	U	U	U

Bit7(RST) 复位类型

- 0: 睡眠后从复位脚唤醒;
1: 睡眠后从 IO 口改变唤醒。

Bit6(GP1) 通用读写位。

Bit5(GP0) 通用读写位。

Bit4(T) 超时标志位

- 0: 发生了 WDT 超时;
1: 上电复位或执行 WDTC 指令、SLEP 指令后被置 1。

Bit3(P) 掉电标志位

- 0: 执行了 SLEP 指令;
1: 上电复位或执行 WDTC 指令。

Bit2(Z) 零标志位

- 0: 算术或者逻辑运算的结果不为零;
1: 算术或者逻辑运算的结果为零。

Bit1(DC) 辅助进位/借位位 (ADD, SUB 指令)

- 0: 未发生执行结果的低 4 位向高 4 位进位;

1: 发生了执行结果的低 4 位向高 4 位进位。

Bit0(C) 进位/借位位 (ADD, SUB 指令)

0: 未发生执行结果向高位进位;

1: 发生了执行结果向高位进位。

注: 对于循环 (RLC、RRC) 指令, 该位装载源寄存器的最高位或最低位。

FSR(RAM 选择寄存器) (地址:0x104)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	D7	D6	D5	D4	D3	D2	D1	D0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

Bits 0~7 用于在间址寻址下选择寄存器(地址: 00~FF)。

如果没有使用间址寻址方式, R4 可以用作一个 8 位通用读/写寄存器。

P5D (端口 5 寄存器) (地址:0x105)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P57	P56	P55	P54	P53	P52	P51	P50
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

P6 D(端口 6 寄存器) (地址:0x106)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P67	P66	P65	P64	P63	P62	P61	P60/EINT
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

P7 D(端口 7 寄存器) (地址:0x107)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol			P75	P74	P73	P72	P71	P70
R/W/C			R/W	R/W	R/W	R/W	R/W	R/W
RESET			0	0	0	0	0	0
WDT_out			0	0	0	0	0	0
Wake_up			U	U	U	U	U	U

P5S (端口 5 状态) (地址:0x115)

TX322 8-BIT MCU

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P57	P56	P55	P54	P53	P52	P51	P50
R/W/C	R	R	R	R	R	R	R	R

P6S (端口 6 狀態) (地址:0x116)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P67	P66	P65	P64	P63	P62	P61	P60
R/W/C	R	R	R	R	R	R	R	R

P7S (端口 7 狀態) (地址:0x117)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol			P75	P74	P73	P72	P71	P70
R/W/C			R	R	R	R	R	R

WUCON6 (P6 喚醒控制寄存器) (地址:0x120)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	WU67	WU66	WU65	WU64	WU63	WU62	WU61	WU60
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

IO 口输入改变唤醒控制位

Bit7(WU67) P67 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit6(WU66) P66 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit5(WU65) P65 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit4(WU64) P64 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit3(WU63) P63 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit2(WU62) P62 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit1(WU61) P61 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit0(WU60) P60 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

WUCON5 (P5 唤醒控制寄存器) (地址:0x121)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	WU57	WU56	WU55	WU54	WU53	WU52	WU51	WU50
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

IO 口输入改变唤醒控制位

Bit7(WU57) P57 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit6(WU56) P56 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit5(WU55) P55 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit4(WU54) P54 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit3(WU53) P53 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit2(WU52) P52 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit1(WU51) P51 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

Bit0(WU50) P50 口输入改变唤醒控制位

0: 禁止唤醒

1: 使能唤醒

OPEN_DRAIN (集电极开路控制寄存器) (地址:0x123)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	OD67	OD66	OD65	OD64	-	OD62	OD61	OD60
R/W/C	R/W	R/W	R/W	R/W	-	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

集电极开路使能，分别控制各个端口的集电极开路是否使能，为 1 使能。

Bit7(OD67) P67 集电极开路控制位

0: 关闭 P67 的集电极开路；

1: 打开 P67 的集电极开路。

Bit6(OD66) P66 集电极开路控制位

0: 关闭 P66 的集电极开路；

1: 打开 P66 的集电极开路。

Bit5(OD65) P65 集电极开路控制位

0: 关闭 P65 的集电极开路；

1: 打开 P65 的集电极开路。

Bit4(OD64) P64 集电极开路控制位

0: 关闭 P64 的集电极开路；

1: 打开 P64 的集电极开路。

Bit2(OD62) P62 集电极开路控制位

0: 关闭 P62 的集电极开路；

1: 打开 P62 的集电极开路。

Bit1(OD61) P61 集电极开路控制位

0: 关闭 P61 的集电极开路；

1: 打开 P61 的集电极开路。

Bit0(OD60) P60 集电极开路控制位

0: 关闭 P60 的集电极开路；

1: 打开 P60 的集电极开路。

IOC5 (P5 端口控制寄存器) (地址:0x125)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P57	P56	P55	P54	P53	P52	P51	P50

R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

“1”设置相关 I/O 管脚为输入脚；

”0”设置相关 I/O 管脚为输出脚。

IOC6 (P6 端口控制寄存器) (地址:0x126)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P67	P66	P65	P64	P63	P62	P61	P60
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

“1”设置相关 I/O 管脚为输入脚；

”0”设置相关 I/O 管脚为输出脚。

IOC7 (P7 端口控制寄存器) (地址:0x127)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol				P74/PWM	P73	P72	P71	P70
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

“1”设置相关 I/O 管脚为输入脚；

”0”设置相关 I/O 管脚为输出脚。

PULL_P5H (P57~P54 上拉下拉控制寄存器) (地址:0x12A)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	/PH57	/PH56	/PH55	/PH54	/PD57	/PD56	/PD55	/PD54
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

上拉或下拉使能，分别控制各个端口的下拉电阻是否使能，为 0 使能。

Bit7(/PH57) P57 上拉控制位

0: 打开 P57 的上拉；

1: 关闭 P57 的上拉。

Bit6(/PH56) P56 上拉控制位

0: 打开 P56 的上拉；

1: 关闭 P56 的上拉。

Bit5(/PH55) P55 上拉控制位

0: 打开 P55 的上拉;

1: 关闭 P55 的上拉。

Bit4(/PH54) P54 上拉控制位

0: 打开 P54 的上拉;

1: 关闭 P54 的上拉。

Bit3(/PD57) P57 下拉控制位

0: 打开 P57 的下拉;

1: 关闭 P57 的下拉。

Bit2(/PD56) P56 下拉控制位

0: 打开 P56 的下拉;

1: 关闭 P56 的下拉。

Bit1(/PD55) P55 下拉控制位

0: 打开 P55 的下拉;

1: 关闭 P55 的下拉。

Bit0(/PD54) P54 下拉控制位

0: 打开 P54 的下拉;

1: 关闭 P54 的下拉。

PULL_DOWN (下拉控制寄存器) (地址:0x12B)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	/PD63	/PD62	/PD61	/PD60	/PD53	/PD52	/PD51	/PD50
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

下拉使能，分别控制各个端口的下拉电阻是否使能，为 0 使能。

Bit7(/PD63) P63 下拉控制位

0: 打开 P63 的下拉;

1: 关闭 P63 的下拉。

Bit6(/PD62) P62 下拉控制位

0: 打开 P62 的下拉;

1: 关闭 P62 的下拉。

Bit5(/PD61) P61 下拉控制位

0: 打开 P61 的下拉;

1: 关闭 P61 的下拉。

Bit4(/PD60) P60 下拉控制位

0: 打开 P60 的下拉;

1: 关闭 P60 的下拉。

Bit3(/PD53) P53 下拉控制位

0: 打开 P53 的下拉;

1: 关闭 P53 的下拉。

Bit2(/PD52) P52 下拉控制位

0: 打开 P52 的下拉;

1: 关闭 P52 的下拉。

Bit1(/PD51) P51 下拉控制位

0: 打开 P51 的下拉;

1: 关闭 P51 的下拉。

Bit0(/PD50) P50 下拉控制位

0: 打开 P50 的下拉;

1: 关闭 P50 的下拉。

T1CON (Timer1 控制寄存器) (地址:0x12C)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol						TM1E	TM1P1	TM1P0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

Bit2(TM1E): TIMER1 是否使能

0: 禁止

1: 允许

Bit1~Bit0(TM1P1-0): TIMER1 预分频比选择 (对 Fosc/4 进行预分频)

TM1P1	TM1P0	分频
0	0	1:1
0	1	1:4
1	0	1:8
1	1	1:16

PULL_HIGH (上拉控制寄存器) (地址:0x12D)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	/PH67	/PH66	/PH65	/PH64	/PH63	/PH62	/PH61	/PH60
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1

Wake_up	U	U	U	U	U	U	U	U
---------	---	---	---	---	---	---	---	---

上拉使能，分别控制各个端口的上拉电阻是否使能，为 0 使能。

Bit7(/PH67) P67 上拉控制位

0: 打开 P67 的上拉；

1: 关闭 P67 的上拉。

Bit6(/PH66) P66 上拉控制位

0: 打开 P66 的上拉；

1: 关闭 P66 的上拉。

Bit5(/PH65) P65 上拉控制位

0: 打开 P65 的上拉；

1: 关闭 P65 的上拉。

Bit4(/PH64) P64 上拉控制位

0: 打开 P64 的上拉；

1: 关闭 P64 的上拉。

Bit3(/PH63) P63 上拉控制位

0: 打开 P63 的上拉；

1: 关闭 P63 的上拉。

Bit2(/PH62) P62 上拉控制位

0: 打开 P62 的上拉；

1: 关闭 P62 的上拉。

Bit1(/PH61) P61 上拉控制位

0: 打开 P61 的上拉；

1: 关闭 P61 的上拉。

Bit0(/PH60) P60 上拉控制位

0: 打开 P60 的上拉；

1: 关闭 P60 的上拉。

WDTCN (WDT 控制寄存器) (地址:0x12E)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	EIS	EI_ES	WDTE	SLPC	WDT_CKS			WU7
R/W/C	R/W	R/W	R/W	R/W	R/W			R/W
RESET	0	0	1	1	0			0
WDT_out	0	0	1	1	U			U
Wake_up	U	U	1	1	U			U

Bit7(EIS) 外部中断选择脚

0: 为 P60 通用 IO 口

1: 为外部中断脚

Bit6(EI_ES) 外部中断触发选择

0: INT 中断脚在下降沿时中断

1: INT 中断脚在上升沿时中断

Bit5(WDTE) WDT 使能位，只有在 OPTION CODE 中的 ENWDT 位为 1 时才起作用

0: WDT 禁止

1: WDT 允许

WDT 控制位。WDTE 位仅在代码选项位 ENWDT 为“0”时无效。也就是说，如果 ENWDT 位为“0”，不管 WDTE 位如何设置，WDT 都是无效。WDTE 位仅在代码选项位 ENWDT 为“1”时有效。如果 ENWDT 代码选项位为“1”，则 WDT 由 WDTE 位设置为无效/有效。0=WDT 无效，1=WDT 有效。WDTE 位是可读可写的。

Bit4(SLPC) SLEEP2 模式控制，当写 0 时进入 SLEEP2 模式。

此位由硬件在唤醒信号的下降沿置 1，由软件进行清 0。SLPC 用于控制振荡器振荡。当由高到低变化时，振荡器停振(振荡器停止工作，控制器进入 SLEEP2 模式)。当由低到高变化时，振荡器起振(控制器从 SLEEP2 模式唤醒)。为了确保振荡器的输出稳定，一旦振荡器从停振到再次起振，需要大约 18ms（振荡器建立时间，OST）的延迟，再执行下一条程序指令。OST 在从休眠模式唤醒时就被激活，不管代码选项位 ENWDT 是否设置为“0”，OST 都会唤醒。如果代码选项位 ENWDT 为“1”，WDT 在唤醒后有效。SLPC 位是可读可写位。

Bit3(WDT_CKS) WDT 时钟源选择位

0: 内部 14K RC 振荡

1: 内部系统主频

Bit0(WU7) P70~P75 IO 口改变唤醒使能位

0: 禁止 P70~P75 IO 口改变唤醒

1: 允许 P70~P75 IO 口改变唤醒

INTIE（中断允许寄存器）（地址:0x12F）

INTIE 寄存器是可读写的寄存器，包含 TMR0 溢出、INT 下降沿外部中断、及 TMR1 匹配中断等各种使能控制位。

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	WDT_IE			ICIE	TM1IE		EXIE	TCIE
R/W/C	R/W			R/W	R/W		R/W	R/W
RESET	0			0	0		0	0
WDT_out	0			0	0		0	0
Wake_up	U			U	U		U	U

Bit7(WDT_IE) 看门狗溢出中断允许

0: 看门狗溢出中断禁止

1: 看门狗溢出中断允许

Bit4(ICIE) IO 口改变中断允许

0: IO 口改变中断禁止

1: IO 口改变中断允许

Bit3(TM1IE) TIMER1 中断允许

0: TIMER1 和 PWP 匹配中断禁止

1: TIMER1 和 PWP 匹配中断允许

Bit1(EXIE) 外部中断允许

0: 外部中断禁止

1: 外部中断允许

Bit0(TCIE) TCC 中断允许

0: TCC 溢出中断禁止

1: TCC 溢出中断允许

- 下列例子 P60 为外部中断脚，每进一次外部中断(P60high 变 low)时改变 P50(1、0 交替)。

```

3  ORG    0X0000
4  JMP    _START
5  ORG    0X0005 ;外部INT中断入口地址
6  JMP    _EXINT
7
8  _EXINT:
9  MOV    A, @0X01
10 XOR    P5D, A
11 BC     INTIF, 1 ;标志清除
12 RETI
13
14 ORG    0X0100
15 _START:
16 CALL   _INIT_RAM
17 CALL   _INIT_IO ;P60设为外部中断脚input pull high, P50 output low
18 MOV    A, @0XB0
19 MOV    WDTCON, A ;P60为外部中断脚，外部中断下降沿触发
20
21 MOV    A, @0X02
22 MOV    INTIE, A ;外部中断允许
23 ENI    ;中断开启
24
25 ...
26

```

- 下列例子 P60 为唤醒脚，进入睡眠时(Sleep2)使用 P60 唤醒，唤醒后执行下一行使 P50 改变(1、0 交替)。

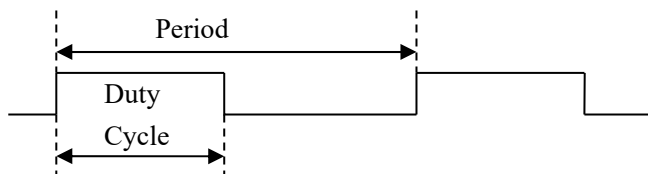
```

2  ORG    0X0000
3  JMP    _START
4  ORG    0x0007    ;IO口改变唤醒中断入口地址
5  JMP    _INT_WPIN
6
7  _INT_WPIN:
8      CLR    INTIF    ;标志清除
9      RETI
10
11 ORG      0X0100
12 _START:
13     CALL    _INIT_RAM
14     CALL    _INIT_IO    ;P60设为唤醒脚input pull high, P50 ouput low
15
16     MOV     A, @0x10
17     MOV     WDTCON, A    ;WDT禁止
18
19     MOV     A, @0x10
20     MOV     INTIE, A    ;ICIE IO口改变中断允许
21     BS      WUCON6, 0    ;P60口输入改变唤醒使能
22     ENI      ;中断开启
23
24     BC      WDTCON, 4    ;进Sleep2睡眠, 等P60唤醒后执行下一行
25     MOV     A, @0x01
26     XOR     P5D, A
27     JMP     $-2

```

PWM1_DC (PWM 占空比寄存器) (地址:0x130)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	DAT7	DAT6	DAT5	DAT4	DAT3	DAT2	DAT1	DAT0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U



$$\text{Duty Cycle} = (\text{PWM_DC} / \text{PWM_PR}) * 100\%$$

PWM2_DC (PWM2 占空比寄存器) (地址:0x131)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	DAT7	DAT6	DAT5	DAT4	DAT3	DAT2	DAT1	DAT0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0

Wake_up	U	U	U	U	U	U	U	U
---------	---	---	---	---	---	---	---	---

PWM1_PR (PWM 周期寄存器) (地址:0x132)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	DAT7	DAT6	DAT5	DAT4	DAT3	DAT2	DAT1	DAT0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

PWM2_PR (PWM2 周期寄存器) (地址:0x133)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	DAT7	DAT6	DAT5	DAT4	DAT3	DAT2	DAT1	DAT0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

PWMICON (PWM1、2 控制寄存器) (地址:0x134)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	PEN2	PEN1	PS2[2:0]			PS1[2:0]		
R/W/C	R/W	R/W	R/W			R/W		
RESET	0	0	000			000		
WDT_out	0	0	000			000		
Wake_up	U	U	U			U		

Bit7(PEN2) PWM2 输出使能位。

0: 禁止 PWM2 输出

1: 允许 PWM2 输出

Bit6(PEN1) PWM1 输出使能位。

0: 禁止 PWM1 输出

1: 允许 PWM1 输出

Bit[5:3](PS2[2:0]) PWM2 时钟预分频 (注: 是对系统频率再分频, 而不是振荡频率)

Bit[2:0](PS1[2:0]) PWM1 时钟预分频 (注: 是对系统频率再分频, 而不是振荡频率)

注意: PWM 的使能, 还需要将 PTMR 打开, 否则输出不了 PWM。

PS[2]	PS[1]	PS[0]	Clock (Hz)	Period
0	0	0	Fsys/1	
0	0	1	Fsys /2	
0	1	0	Fsys /4	
0	1	1	Fsys /8	
1	0	0	Fsys /16	

1	0	1	Fsys /32	
1	1	0	Fsys /64	
1	1	1	Fsys /128	

若系统设为二分频, 8 bits PWM 频率范围

最快 : PWM 频率 = $F_{osc} / 2 / 255 = F_{osc} / 510$,

Duty=0, 1/255, 2/255, ..., 255/255

最慢 : PWM 频率 = $F_{osc} / 256 / 255 = F_{osc} / 65280$,

Duty=0, 1/255, 2/255, ..., 255/255

PWM 的计算, 以下以 PWM1 举例:

PWM 频率 = (振荡频率 / PWM1 时钟分频) / PWM1_PR 的值

占空比 = (PWM_DC / PWM_PR) * 100%

例如 : 振荡频率 = 455KHz 预分频为 0 即 2 分频 (假设 $F_{sys} = F_{osc} / 2$)

PWM1_PR = 6 PWM1_DC = 2

PWM 频率 = $455K / 2 / 6 = 37.9KHz$

占空比 = $2 / 6 * 100\% = 33.3\%$

PTMR1CON (PTMR1/PTMR2 控制寄存器) (地址:0x135)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	PTM1E	PTM1IE	PTM1IF	PTM2E	PTM2IE	PTM2IF	PO_INV	PTM12CA
R/W/C	R/W	R/W	R/C	R/W	R/W	R/C	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

Bit7(PTM1E): PTIMER1 是否使能

0: 禁止 PTIMER1

1: 打开 PTIMER1

Bit6(PTM1IE): PTIMER1 中断是否使能

0: 禁止 PTIMER1 中断

1: 使能 PTIMER1 中断

Bit5(PTM1IF): PTIMER1 中断标志

0: 没有 PTIMER1 中断

1: 有 PTIMER1 中断

Bit4(PTM2E): PTIMER2 是否使能

0: 禁止 PTIMER2

1: 打开 PTIMER2

Bit3(PTM2IE): PTIMER2 中断是否使能

0: 禁止 PTIMER2 中断

1: 使能 PTIMER2 中断

Bit2(PTM2IF): PTIMER2 中断标志

0: 没有 PTIMER2 中断

1: 有 PTIMER2 中断

Bit1(PO_INV): PWM 输出是高有效还是低有效选择

0: 平时为低, PWM 输出高有效

1: 平时为高, PWM 输出低有效

Bit0(PTM12CAS): PTIMER1 和 PTIMER2 8 位模式与 16 位模式选择

0: 独立的 8 位模式

1: 层叠的 16 位模式

(1) 16bits PWM 的周期寄存器 MSB=PWM2_PR, LSB=PWM1_PR.

(2) 16bits PWM 的占空比寄存器 MSB=PWM2_DC, LSB=PWM1_DC.

(3) 16bits PWM, Period (PWM16_PR) MSB=PWM2_PR, LSB=PWM1_PR.

Duty (PWM16_DC) MSB=PWM2_DC, LSB=PWM1_DC.

PWM16 频率= (振荡频率 / PWM1 时钟分频) / PWM16_PR 的值

占空比= (PWM16_DC / PWM16_PR) * 100%

若系统设为二分频, 16bits PWM 频率范围

最快 : PWM16 频率= $F_{osc} / 2 / (256^2 - 1) = F_{osc} / 131070$,

Duty=0, 1/65535, 2/65535, ..., 65535/65535

最慢 : PWM16 频率= $F_{osc} / 256 / (256^2 - 1) = F_{osc} / 16776960$,

Duty=0, 1/65535, 2/65535, ..., 65535/65535

(4) (PWM2_PR, PWM1_PR) >= (PWM2_DC, PWM1_DC)

- 以输出 PWM1 站控比 20%, PWM2 站控比 50% 为例

```

10  ...
11  MOV  A, @20
12  MOV  PWM1_DC, A
13  MOV  A, @100
14  MOV  PWM1_PR, A
15
16  MOV  A, @50
17  MOV  PWM2_DC, A
18  MOV  A, @100
19  MOV  PWM2_PR, A
20
21  MOV  A, @0x00
22  MOV  IO_FUNC, A ;设置PWM输出脚 PWM1=P61, PWM2=P62
23
24  MOV  A, @0xC9
25  MOV  PWM1CON, A ;PWM1,2 输出使能, 时钟预分频=FSYS/2
26
27  MOV  A, @0x90
28  MOV  PTMR1CON, A ;PTIMER1,2 使能(务必开启 开启后pwm才会动作),
29
30  ...

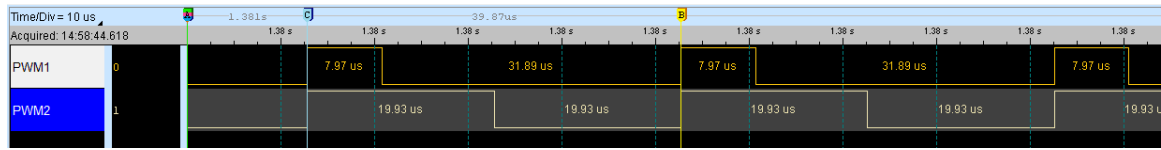
```

Fosc= 10M

$$F_{sys} = 10M/2 = 5M$$

$$\text{预分频} = 5M/2 = 2.5M$$

$$\text{PWM 频率} = (2.5M/100) = 25kHz(40\mu s)$$



- 下述例子为 PTM12CAS 16 模式(PWM2 输出)

```

10  ...
11  MOV  A, @20
12  MOV  PWM1_DC, A
13  MOV  A, @100
14  MOV  PWM1_PR, A
15
16  MOV  A, @50
17  MOV  PWM2_DC, A
18  MOV  A, @100
19  MOV  PWM2_PR, A
20
21  MOV  A, @0x00
22  MOV  IO_FUNC, A ;设置PWM输出脚 PWM1=P61, PWM2=P62
23
24  MOV  A, @0xC9
25  MOV  PWM1CON, A ;PWM1,2 输出使能, 时钟预分频=FSYS/2
26
27  MOV  A, @0x91
28  MOV  PTMR1CON, A ;PTIMER1,2 使能(务必开启 开启后pwm才会动作),
29                      ;PTM12CAS 16位模式 由PWM2 输出
30  ...

```

$$\text{PWM 16DC} = (\text{MSB}=\text{PWM2_DC}=50, \text{LSB}=\text{PWM1_DC}=20) = 0x3214 = 12820$$

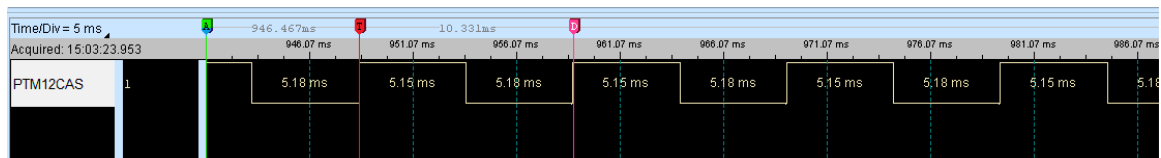
$$\text{PWM 16 PR} = (\text{MSB}=\text{PWM2_PR}=100, \text{LSB}=\text{PWM1_PR}=100) = 0x6464 = 25700$$

$$\text{占空比} = (12820 / 25700) * 100\% = 0.49\%$$

$$F_{osc} = 10M, F_{sys} = 10M/2 = 5M$$

$$\text{预分频} = 5M/2 = 2.5M$$

$$\text{PTM12CAS 频率} = (2.5M/25700) = 97.27Hz(10.28ms)$$



TMR1 (TIMER1 寄存器) (地址:0x13E)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	TMR17	TMR16	TMR15	TMR14	TMR13	TMR12	TMR11	TMR10
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TX322 8-BIT MCU

RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

该寄存器中为 TIMER1 计数值，当计到和 PWP 相等时重新从 00 开始。

PWP (脉宽预置寄存器) (地址:0x13F)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	PWP7	PWP6	PWP5	PWP4	PWP3	PWP2	PWP1	PWP0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

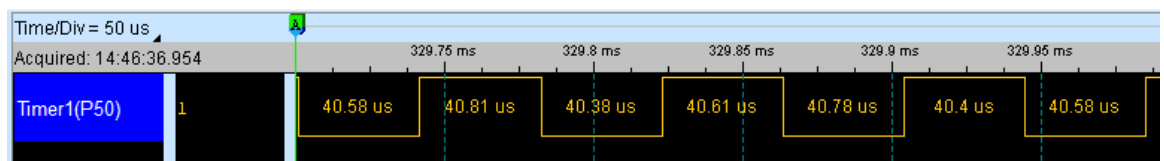
PWP7~ PWP0 设置成与波特率时钟期望宽度。

- 以下例子为 TIMER1 从 0 开始，数到 100 后进中断改变 P50 状态(1、0 交替)，中断的频率为 25khz。

```

3  ORG    0x0000
4  JMP    _START
5  ORG    0x0011 ;PWP (TIMER1) 中断入口地址
6  JMP    _INT_TIMER1
7
8  _INT_TIMER1:
9  MOV    A, @0x01
10 XOR    P5D, A
11 BC     INTIF, 3 ;TIMER1中断标志清除
12 RETI
13
14 ORG    0x0100
15 _START:
16 CALL   _INIT_IO ;设置I/O管脚, P5 output low
17
18 MOV    A, @0x04
19 MOV    T1CON, A ;TIMER1使能, 预分频1:2
20
21 MOV    A, @0x08
22 MOV    INTIE, A ;TIMER1中断允许
23 MOV    A, @100
24 MOV    PWP, A ;PWP设100
25
26 ENI ;中断开启
27
28 NOP
29 JMP    $-1

```



$F_{osc} = 10M$, $F_{sys} = 10M/2 = 5M$

预分频 = 1:2

TIMER1 中断时间 = $100((1/5M) * 2) = 40\mu s$

PAL (表格指针低 8 位) (地址: 0x14B)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	A7	A6	A5	A4	A3	A2	A1	A0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	X	X	X	X	X	X	X	X
WDT_out	X	X	X	X	X	X	X	X
Wake_up	U	U	U	U	U	U	U	U

PAH (表格指针高 8 位) (地址: 0x14C)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	A15	A14	A13	A12	A11	A10	A9	A8
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	X	X	X	X	X	X	X	X
WDT_out	X	X	X	X	X	X	X	X
Wake_up	U	U	U	U	U	U	U	U

此查表方法可在程序存储器的任意空间使用，不受任何限制，表格指针必须先行设定，其方式是将表格的低 7 位地址放在表格指针寄存器 PAL[7:1]中。PAL[0]用来表示表格的高低字节，PAL[0]为 0 是低字节，PAL[0]为 1 是高字节。将表格的高 6 位地址放在表格指针寄存器 PAH[5:0]，这个寄存器存放表格较高的 6 位地址。在设定完表格指针后，表格数据可以使用“MOVC”指令从当前程序所在的存储器中来查表读取。

例：

CODE_TAB:

```

MOV     PAL, A           ; 将偏移量送入低 8 位指针
MOV     A, @(CODE_TAB_DB >> 7) ; 取表格指针地址的高 6
                                ; 位地址
MOV     PAH, A           ; 并送入表格指针的高位地址
MOV     A, @((CODE_TAB_DB << 1) & 0X00FF) ; 取表格指针地址
                                ; 的低 7 位送到 PAL[7:1]中
ADD     PAL, A           ; 加偏移量并送入低 8 位指针中
MOV     A, @0            ; 判断是否有进位
JBC     STATUS, C        ; 有进位就加 1
MOV     A, @1            ; 没有进位就加 0
ADD     PAH, A           ;
MOVC    ; 读数查表
RET

```

CODE_TAB_DB: ; 低位 8 位数据在前

DB 0X014, 0X00F ;K1,K2

DB 0X00B, 0X00C ;K3,K4

DB 0X00E, 0X016 ;K5,K6
DB 0X000, 0X001 ;K7,K8

UA_CON (UART 控制寄存器) (地址:0x150)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	UA_EN	RX_EN	RX_IE	TX_IE	BAUD16	TXOF/TXF	RXF	RX_OV
R/W/C	R/W	R/W	R/W	R/W	R/W	R/C	R	R/C
RESET	0	0	0	0	0	0	0	0

Bit7(UA_EN) UART 模块使能位

- 0: 禁止 UART 模块
- 1: 使能 UART 模块

Bit6(RX_EN) UART 接收模块使能位

- 0: 禁止 UART 的接收模块
- 1: 使能 UART 的接收模块

Bit5(RX_IE) UART 接收模块中断使能位

- 0: 禁止 UART 接收模块的中断
- 1: 使能 UART 接收模块的中断

Bit4(TX_IE) UART 发送模块中断使能位

- 0: 禁止 UART 发送模块的中断
- 1: 使能 UART 发送模块的中断

Bit3(BAUD16) UART 的波特率主频设置位

- 0: UART 的波特率主频为 $F_{sys}/2$
- 1: UART 的波特率主频为 $F_{sys}/8$

Bit2(TXOF/TXF) UART 的发送中断标志和发送标志

- 0: 当有开中断时，需要软件清零，若没有开中断，则为发送的空闲状态
- 1: 当有开中断时，为发送完成的标志，需软件清零；若没有开中断，则为发送状态标志，表示正在发送，发送完后为 0.

Bit1(RXF) UART 接收标志位

- 0: 表示接收空闲状态
- 1: 表示正在接收

Bit0(RX_OV) UART 接收完成标志位

- 0: 需软件清零
- 1: UART 接收完成标志

UA_BAUD (UART 的波特率设置寄存器) (地址:0x151)

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

Symbol	BAUD7	BAUD6	BAUD5	BAUD4	BAUD3	BAUD2	BAUD1	BAUD0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	U	U	U	U	U	U	U	U

波特率值的计算如下：

当 UA_CON.BAUD16=1 时，波特率值= $(F_{sys}/8) / (BAUD + 1)$, $BAUD > 0$.

当 UA_CON.BAUD16=0 时，波特率值= $(F_{sys}/2) / (BAUD + 1)$, $BAUD > 1$.

例如：

振荡 $F_{sys} = 12\text{MHz}/2$ ，假设波特率 = 115200Hz., UA_CON.BAUD16=0 时

$BAUD = (12\text{M} / 2 / 2) / 115200 - 1 = 12 = 0x19$

UA_BUF (UART 的数据寄存器) (地址:0x152)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	BUF7	BUF6	BUF5	BUF4	BUF3	BUF2	BUF1	BUF0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	U	U	U	U	U	U	U	U

此寄存器为 UART 发送和接收的数据寄存器。

注意：我们的 UART 为半双工 UART

- 透过 UART 循环传送 4 笔数据(0x55, 0xAA, 0x00, 0xFF)，预设波特率值 9600。(注意以下例子为同一片 IC TX 跟 RX 对接)。

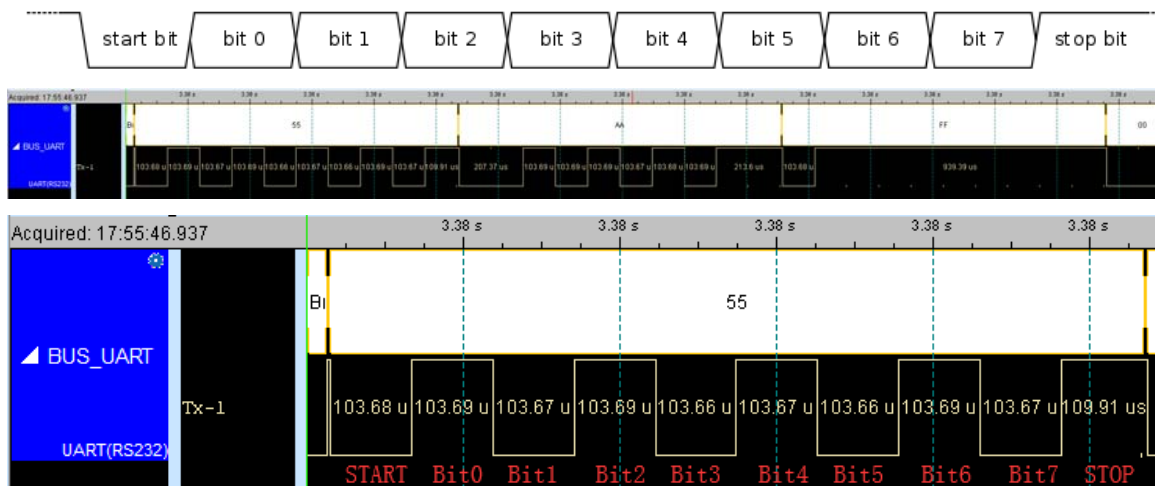
```

10 ORG    0x0000
11 JMP    _START
12 ORG    0x001D ;UART中断入口地址
13 BC     UA_CON, 2 ;发送中断标志和发送标志清除
14 BC     UA_CON, 0 ;UART接收完成标志清除
15 RETI
16
17 _START:
18 CALL   _INIT_RAM
19 CALL   _INIT_IO ;设置I/O管脚 ,P72=TX 设为 ouput, P73=RX 设为 input
20
21 MOV    A, @0xE0
22 MOV    UA_CON, A ;UART模块使能,UART接收模块 及 接收模块中断使能,Baud16=0
23
24 MOV    A, @103
25 MOV    UA_BAUD, A ;UART 的波特率9600=(Fsys/2)/(UA_BAUD+1)
26
27 BS     P7D, 2 ;UART TX 初始要在High
28 ENI ;中断开启
29
30 _LOOP:
31 INC    COUNT1
32 MOV    A, COUNT1
33 CALL   _DATA_TBL
34 UART_WRITE:
35 MOV    UA_BUF, A ;UART传送_DATA_TBL的值
36 MOV    DATA, A
37
38 JBC    UA_CON, 2 ;判断TX是否傳送完成
39 JMP    $-1
40 ;接收完成后进中断,清除标志
41 _UART_READ:
42 MOV    A, UA_BUF ;判断接收的与传送的值是否相同
43 XOR    A, DATA
44 JBS    STATUS, Z
45 JMP    _ERR ;若不相同执行_ERR
46
47 MOV    A, @0x04
48 XOR    A, COUNT1
49 JBC    STATUS, Z
50 CLR    COUNT1
51 JMP    _LOOP ;下一笔资料
52
53 _DATA_TBL:
54 TBL
55 NOP
56 RETL    @0x55
57 RETL    @0xAA
58 RETL    @0xFF
59 RETL    @0x00
60
61 _ERR:
62 BS     P5D, 1
63 JMP    $-1

```

TX 起始高电位，start bit 时低电位，stop bit 时高电位。

TX322 8-BIT MCU



$F_{osc} = 4M$, $F_{sys} = 4M/2 = 2M$

UA_CON.BAUD16=0 时，波特率值 = $(F_{sys} / 2) / (BAUD + 1)$

波特率值=9600

$BAUD = (2M/2)/9600 - 1 = 103$

IO_PULL2 (上拉控制寄存器 2) (地址:0x15F)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol					/P7_PH			
R/W/C					R/W			
RESET					1			
WDT_out					1			
Wake_up					U			

上拉使能，分别控制各个端口的上拉电阻是否使能，为 0 使能。

Bit3(/P7_PH) P75~P70 上拉控制位

0: 打开 P75~P70 的上拉;

1: 关闭 P75~P70 的上拉。

IO_SINK2 (IO SINK2 设定寄存器) (地址:0x171)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol					P7_XS			P5H_XS
R/W/C					R/W			R/W
RESET					0			0
WDT_out					0			0
Wake_up					U			U

Bit3(P7_XS) 当 P75~P70 口设成输出，使能该 IO 口 Sink 能力。

0: 正常输出

1: 加强 P75~P70 的 Sink 输出

Bit0(P5H_XS) 当 P57~P54 口设成输出，使能该 IO 口 Sink 能力。

0: 正常输出

1: 加强 P57~P54 的 Sink 输出

IO_FUNC (IO 特殊功能设定寄存器) (地址:0x172)

	Bit 7	Bit 6	Bit 5	Bit 4-3	Bit 2	Bit 1-0
Symbol				F_CKO [1:0]	F_UART	F_PWM[1:0]
R/W/C				R/W	R/W	R/W
RESET				00	0	00
WDT_out				00	0	00
Wake_up				U	U	U

Bit4-3(F_CKO) 设定振荡从 P66 除频输出

00: P66

01: Fosc

10: Fosc / 2

11: Fosc / 4

Bit2(F_UART) 设定 UART 的功能脚

0: RX=P64; TX=P65

1: RX=P73; TX=P72.

Bit1-0(F_PWM) 选择 PWM 输出脚

00: PWM1 = P61, PWM2 = P62

01: PWM1 = P64, PWM2 = P65

10: PWM1 = P66, PWM2 = P67

11: 保留

IO_BUF2 (IO Buffer 设定寄存器) (地址:0x173)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol					P7_ST			P5H_ST
R/W/C					R/W			R/W
RESET					1			1
WDT_out					1			1
Wake_up					U			U

Bit3(P7_ST) 设定 P75~P70 的 IO 缓冲区

0: P75~P70 为 CMOS 输入

1: 施密特输入, 当做输出时有串 330 的电阻

Bit0(P5H_ST) 设定 P57~P54 的 IO 缓冲区

0: P57~P54 为 CMOS 输入

1: P57~P54 为施密特输入, 当做输出时有串 330 的电阻

IO_SINK (IO SINK 设定寄存器) (地址:0x174)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P523_XS	P501_XS	P67_XS	P66_XS	P645_XS	P623_XS	P61_XS	P60_XS
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TX322 8-BIT MCU

RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

Bit7(P523_XS) 当 P52, P53 口设成输出, 使能该 IO 口的 Sink 能力.

0: P52 与 P53 为正常输出

1: 加强 P52 与 P53 的 Sink 输出

Bit6(P501_XS) 当 P50, P51 口设成输出, 使能该 IO 口 Sink 能力.

0: P50 与 P51 为正常输出

1: 加强 P50 与 P51 的 Sink 输出

Bit5(P67_XS) 当 P67 口设成输出, 使能该 IO 口 Sink 能力.

0: P67 为正常输出

1: 加强 P67 的 Sink 输出

Bit4(P66_XS) 当 P66 口设成输出, 使能该 IO 口 Sink 能力.

0: P66 为正常输出

1: 加强 P66 的 Sink 输出

Bit3(P645_XS) 当 P64, P65 口设成输出, 使能该 IO 口 Sink 能力.

0: P64 与 P65 为正常输出

1: 加强 P64 与 P65 的 Sink 输出

Bit2(P623_XS) 当 P62, P63 口设成输出, 使能该 IO 口 Sink 能力.

0: P62 与 P63 为正常输出

1: 加强 P62 与 P63 的 Sink 输出

Bit1(P61_XS) 当 P61 口设成输出, 使能该 IO 口 Sink 能力.

0: P61 为正常输出

1: 加强 P61 的 Sink 输出

Bit0(P60_XS) 当 P60 口设成输出, 使能该 IO 口 Sink 能力.

0: P60 为正常输出

1: 加强 P60 的 Sink 输出

IO_BUF (IO Buffer 设定寄存器) (地址:0x175)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	P523_ST	P501_ST	P67_ST	P66_ST	P645_ST	P623_ST	P61_ST	P60_ST
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

Bit7(P523_ST) 设定 P52, P53 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit6(P501_ST) 设定 P50, P51 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit5(P67_ST) 设定 P67 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit4(P66_ST) 设定 P66 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit3(P645_ST) 设定 P64, P65 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit2(P623_ST) 设定 P62, P63 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit1(P61_ST) 设定 P61 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

Bit0(P60_ST) 设定 P60 buffer

0: CMOS 输入

1: 施密特输入, 在输出时会串一个 330 左右的电阻再输出

OSCCON (OSC 控制寄存器) (地址:0x178)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol		GREEN						
R/W/C		R/W						
RESET		0						
WDT_out		0						
Wake_up		U						

Bit6(GREEN) 绿色模式控制位

0: 睡眠后进入普通模式, 及所有工作都停止

1: 睡眠后进入绿色模式, 指令停止工作, 但 TCC 和 Timer1 及 PWM 还会继续工作

注意: 此位只能用 BC 或 BS 操作

GCKCON (功能模块使能控制寄存器) (地址:0x17C)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	G_SYS	G_GPIO	G_TMR1				G_UART	G_PWM
R/W/C	R/W	R/W	R/W				R/W	R/W
RESET	1	1	1				1	1
WDT_out	1	1	1				1	1
Wake_up	U	U	U				U	U

Bit7(G_SYS) 系统模块使能控制位,

可控制的寄存器为: IO_SINK、IO_SINK2、IO_BUF、IO_BUF2、IO_FUNC、LDOCON、LVRCON、和 ACHCON。

0: 禁止系统模块

1: 使能系统模块

Bit6(G_GPIO) IO 模块使能控制位

可控制的寄存器为: WUCON6、WUCON5、PULL_P5H、OPN_DRN、OPN_DRN2、PULL_DOWN、PULL_HIGH、IO_PULL2 和 WDTCON。

0: 禁止 IO 模块

1: 使能 IO 模块

Bit5(G_TMR1) Timer1 模块使能控制位

可控制 Timer1 模块的寄存器为: T1CON、TMR1、PWP

0: 禁止 Timer1 模块、UART 模块

1: 使能 Timer1 模块、UART 模块

Bit1(G_UART) UART 模块使能控制位

可控制的寄存器为 UA_CON, UA_BAUD 和 UA_BUF

0: 禁止 UART 模块

1: 使能 UART 模块

Bit0(G_PWM) PWM 模块使能控制位

可控制的寄存器为: PWMx_DC、PWMx_PR、PWMxCON。

0: 禁止 PWM 模块

1: 使能 PWM 模块

LVRCON (LVR 控制寄存器) (地址:0x17D)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	LVR_EN	LVD_F	LVD_IE	LVR2_EN	FT_EN	LVS2	LVS1	LVS0
R/W/C	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	1	0	1	1	1
WDT_out	0	U	0	1	0	1	1	1
Wake_up	U	U	U	U	U	U	U	U

Bit7(LVR_ENB) 低电压复位使能

0: 使能低电复位

1: 禁止低电压复位

Bit6(LVD_F) 低电压检测标志位

0: 没有低于设定的电压.

1: 检测到有低于设定的电压

Bit5(LVD_IE) 低电压检测中断使能位

0: 中断不使能

1: 中断使能

Bit4(LVR2_EN) 低电压检测使能位

0: 使能低电压检测

1: 禁止低电压检测

注意: 当使能低电压检测时, 会有 20uA 到 30uA 的功耗

当低电压检测使能时, 低电压复位自动使能为 2.0V 复位

Bit3(FT_EN) 低电压复位触发脉宽使能位

0: 没有触发脉宽, 有低电压就复位

1: 有触发脉宽, 触发脉宽大于 960uS

Bit2-0(LVS) 低电压复位和低电压检测的电压选择

当低电压检测没有开时, 低电压复位测 VDDL 的电压, 000 的复位电压为 2.0V

当低电压检测使能后, 低电压复位开启为 2.0V, 低电压检测是测的 VDD 的电压

000 : 4.4v

111 : 2.9V

ISR (INTIF: 中断标志寄存器) (地址:0x17F)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	WDT_IF			ICIF	TM1IF		EXIF	TCIF
R/W/C	R/W			R/W	R/W		R/W	R/W
RESET	0			0	0		0	0
WDT_out	0			0	0		0	0
Wake_up	U			U	U		U	U

读此寄存器的结果为 IOCF 和 ISR 的逻辑与。

Bit7(WDT_IF) WDT 溢出中断标志位

0: 没有 WDT 溢出中断

1: 有 WDT 溢出中断

Bit4(ICIF) IO 口改变中断标志

0: IO 口改变没有中断

1: IO 口改变有中断

Bit3(TM1IF) Timer1 中断标志

0: 没有中断

1: 有中断

Bit1(EXIF) 外部中断标志

0: 外部没有中断

1: 外部有中断

Bit0(TCIF) TCC 中断标志

0: TCC 没有中断

1: TCC 有中断

PWM3_DC (PWM3 占空比寄存器) (地址:0x180)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	DAT7	DAT6	DAT5	DAT4	DAT3	DAT2	DAT1	DAT0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0
WDT_out	0	0	0	0	0	0	0	0
Wake_up	U	U	U	U	U	U	U	U

PWM3_PR (PWM3 周期寄存器) (地址:0x182)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	DAT7	DAT6	DAT5	DAT4	DAT3	DAT2	DAT1	DAT0
R/W/C	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	1	1	1	1	1	1	1	1
WDT_out	1	1	1	1	1	1	1	1
Wake_up	U	U	U	U	U	U	U	U

PWM3CON (PWM3、4 控制寄存器) (地址:0x184)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol		PEN3						PS3[2:0]
R/W/C		R/W						R/W
RESET		0						000
WDT_out		0						000
Wake_up		U						U

Bit6(PEN3) PWM3 输出使能位。

0: 禁止 PWM3 输出

1: 允许 PWM3 输出

Bit[2:0](PS3[2:0]) PWM3 时钟预分频 (注: 是对系统频率再分频, 而不是振荡频率)

PTMR3CON (PTMR3/PTMR4 控制寄存器) (地址:0x185)

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Symbol	PTM3E	PTM3IE	PTM3IF				PO_INV	
R/W/C	R/W	R/W	R/C				R/W	
RESET	0	0	0				0	
WDT_out	0	0	0				0	

Wake_up	U	U	U				U	
---------	---	---	---	--	--	--	---	--

Bit7(PTM3E): PTIMER3 是否使能

0: 禁止 PTIMER3

1: 打开 PTIMER3

Bit6(PTM3IE): PTIMER3 中断是否使能

0: 禁止 PTIMER3 中断

1: 使能 PTIMER3 中断

Bit5(PTM3IF): PTIMER3 中断标志

0: 没有 PTIMER3 中断

1: 有 PTIMER3 中断

Bit4(PTM4E): PTIMER4 是否使能

0: 禁止 PTIMER4

1: 打开 PTIMER4

Bit1(PO_INV): PWM3 输出是高有效还是低有效选择

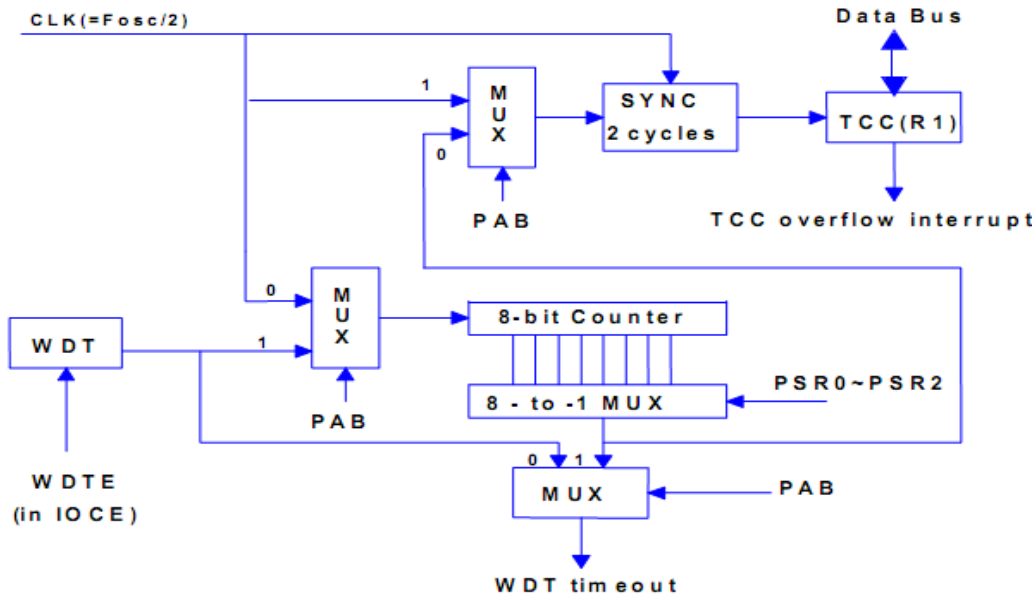
0: 平时为低, PWM3 输出高有效

1: 平时为高, PWM3 输出低有效

五、功能描述

5.1 TCC/WDT 预分频器

有一个 8bit 计数器可以作为 TCC 或 WDT 的分频器。这个预分频器在同一时刻只能提供给 TCC 或 WDT 之一使用, CONT 寄存器的 PAB 位用于决定预分频器通过每次写入 TCC 的指令清除。当分配给 WDT 模式时, WDT 和预分频器由指令 WDTC 和 SLEP 进行清 0。



TCC 和 WDT 的模块图

R1(TCC)是一个 8bit 定时器/计数器。TCC 在每个指令周期(无预分频)加 1。

WDT 是一个自由运行的片内 RC 振荡器。即使是振荡器关闭（例如：在休眠模式），WDT 仍然保持运行。在正常操作或者休眠模式，WDT 溢出（如果有效）将导致复位。在正常模式下，WDT 在任何时候都可以通过软件编程设置为无效或有效（如果代码选项位 ENWDT 为“1”）。参考 IOCE 寄存器的 WDTE 位。没有预分频器时，WDT 的时间输出周期大约是 18MS。

5.2 I/O 端口

I/O 寄存器组，端口 5~端口 7，是双向三态 I/O 端口。

P50~P57、P60~P63 通过软件控制可以具有内部下拉。

P60~P67、P70~P75 通过软件控制可以具有内部上拉。

P60~P67、P70~P75 通过软件控制可以具有唤醒功能。

通过编程控制 I/O 控制寄存器组（IOC5~IOC7），I/O 端口可以被定义为“输入”或“输出”管脚。I/O 寄存器组和 I/O 控制寄存器组都是可读可写的。下图描述了 I/O 接口电路。注意在读取 I/O 端口时输入和输出管脚的读取路径是不同的。

TMR1: 定时器 1 寄存器, TMR1 加 1 直到它与 PWP 值相等,然后清为 0。

比较器: PWP 和 TMR1 相等改变输出状态, 同时将置 TMR1IF 为 1。

- 可编程相关寄存器组

TMR1 的相关状态/数据寄存器

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMR1/RE	TMR17	TMR16	TMR15	TMR14	TMR13	TMR12	TMR11	TMR10
PWP/RF	PWP7	PWP6	PWP5	PWP4	PWP3	PWP2	PWP1	PWP0
T1CON/IOCC	0	0	0	0	0	TM1E	TM1P1	TM1P0

TMR1: 定时器寄存器, TMR17~TMR10 加 1 直到与 PWP 相同, 然后 TMR1 清 0

PWP: 脉冲宽度预置寄存器。PWP7~PWP0 为预先写入波特率时钟期望宽度值

T1CON: TIMER1 控制寄存器

TM1E (BIT2): Timer1 允许位

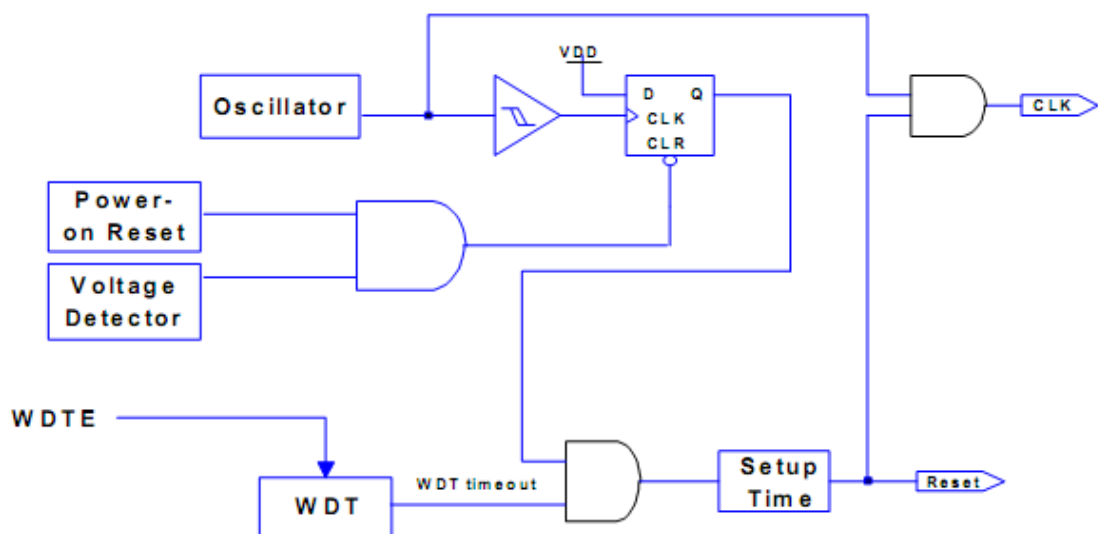
TM1P1 和 TM1P0 (BIT1~0): Timer1 为 FSCO 预分频位

TM1P1	TM1P0	Prescaler Rate
0	0	1:1
0	1	1:4
1	0	1:8
1	1	1:16

5.4 复位和唤醒

复位的原因有:

- (1) 上电复位
- (2) WDT 溢出 (如果有效)



复位电路框图

在检测到复位信号后, 设备将保持一个大约 36ms (振荡器起振时间) 复位时间的状态。一旦复位发生, 以下功能将被执行:

- 振荡器正在运行或者开始运行。
- 程序计数器(R2)设置为全“1”。
- 当电源打开，R3 的 5~6 位和 R4 的高 2 位被清 0。
- 所有 I/O 端口的管脚被设置为输入模式（高阻抗状态）
- WDT 和预分频器清 0。
- 如果代码选项 ENWDT 为“1”，WDT 有效。
- CONT 寄存器的 BIT3、6 清 0，IOCE 寄存器的 BIT0、4~5 设置为“1”。
- 清除 R3F 的 BIT0 和 IOCF 寄存器的 BIT0。

注意：VDD=3V，振荡器起振时间为 $36\text{ms} \pm 30\%$

执行 SLEP 指令（命名为 SLEEP1 模式）可以进入休眠模式（省电模式）。进入休眠模式时，WDT（如果有效）被清除但仍然保持运行。WDT 溢出（如果有效）就唤醒并复位。R3 的 T、P 标志位可判断复位（唤醒）的原因。

除了基本的 SLEEP1 模式，TX3222 还有一种休眠模式 2，由清 IOCE 寄存器的“SLPC”位产生，命名为 SLEEP2 模式。在 SLEEP2 模式下，控制器可由下列情况唤醒：

- (a) 任何一个唤醒管脚为 0 均可唤醒控制器，当唤醒后，控制器将继续执行在线程序。

在进入 SLEEP2 模式之前，触发源（P60~P67）的唤醒功能需要进行设置（例如：输入管脚）并且使其有效（例如：上拉、唤醒控制）。必须注意的是，在唤醒后，如果代码选项位 ENWDT 为“1”，WDT 是有效的。在唤醒后应该通过软件控制 WDT 的操作（设置为无效或有效）。

- (b) WDT 溢出（如果有效）。一旦唤醒，将导致控制器复位。

复位时状态寄存器 T、P 位状态

以下溢出情况，将会引起复位：1.电源上电。2.WDT 溢出。

表 复位后半 T 和 P 值

Reset Type	T	P
Power on	1	1
WDT during Operation mode	0	P
WDT wake-up during SLEEP1 mode	0	0
WDT wake-up during SLEEP2 mode	0	P
Wake-Up on pin change during SLEEP2 mode		

*p:复位前值

表 不同事件对 T、P 影响

Event	T	P
Power on	1	1
WDTC instruction	1	1
WDT time-out	0	*P
SLEEP instruction	1	0
Wake-Up on pin change during SLEEP2 mode	P	P

2.内部 RC 振荡模式

选择内部 RC 振荡模式时，可在烧录时选择 100K、200K、400K、455K、1M、2M、3M、4M、6M、8M、10M、12M、16M、20M 的频率

六、绝对最大范围

(所有电压参考 GND)

项目	符号	额定值	单位
耐压范围	V_{DD}	0 ~6	V
输入/输出电压	V_I / V_O	GND-0.3~VDD+0.3	V
工作温度	T_{DD}	-40 ~ 85	°C
存放温度	T_{ST}	-20 ~ 125	°C

七、电气特性

交流电气特性 ($T_A=0^{\circ}\text{C}\sim 70^{\circ}\text{C}$, $V_{DD}=3\text{V} \& 5\text{V}$)

参数	符号	条件	最小	典型	最大	单位
Input CLK duty cycle	Dclk		45	50	55	%
Instruction cycle time (CLKS= “0”)	Tins	XTAL Type	125		DC	ns
		RC Type	500		DC	ns
TCC input period	Ttcc	Note 1	(Tins+20) /N			ns
Input pin setup time	Tset			0		ns
Input pin hold time	Thold			20		ns
Output pin delay time	Tdelay			50		ns

注意：N=预分频系数

直流电气特性 ($T_A=0^{\circ}\text{C}\sim 70^{\circ}\text{C}$, $V_{DD}=3\text{V} \& 5\text{V}$, LD0 = 3.0V)

符号	参数	测试条件		最小	典型	最大	单位
		VDD	条件				
V_{DD}	Operating Voltage		LDO on 3V $f_{SYS}=4\text{MHz}$ LVR disabled	2.0		6.0	V
			LDO off $f_{SYS}=4\text{MHz}$ LVR disabled	2.0		3.6	
F_{xt}	Supply Oscillator Crystal	3V	One Cycle with two clocks	DC		20	MHz
		5V		DC		20	
I_{IL1}	Input Leakage Current for input pins	3V	$V_{IN} = V_{DD}, V_{SS}$			1	uA
		5V				1	
I_{CC1}	Operating supply current	3V	Fosc = 32KHz, output pin floating, input pins at VDD, WDT disabled		30		uA
		5V			40		

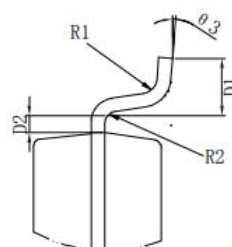
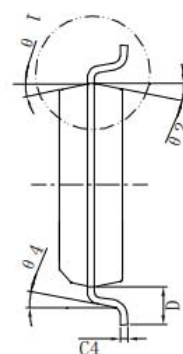
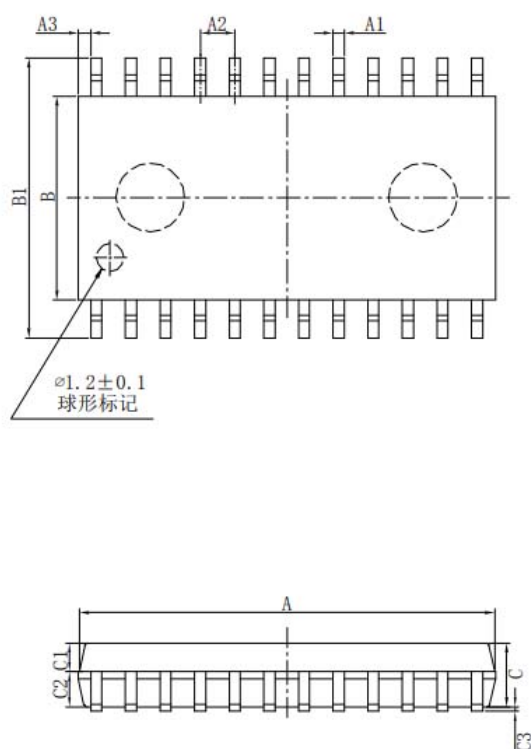
TX322 8-BIT MCU

I _{CC3}	Operating supply current	3V	Fosc = 4MHz, output pin floating, input pins at VDD, WDT disabled		1.3		mA
		5V			2.0		
I _{CC4}	Operating supply current	3V	Fosc = 4MHz, output pin floating, input pins at VDD, WDT enabled		1.3		mA
		5V			2.0		
I _{STB1}	Standby Current	3V	output pin floating, input pins at VDD, WDT disabled, LVR disabled		1		uA
		5V			1		
I _{STB2}	Standby Current	3V	output pin floating, input pins at VDD, WDT enabled, LVR disabled		3		uA
		5V			3		
I _{STB3}	Standby Current	3V	output pin floating, input pins at VDD, WDT disabled, LVR enabled		3		uA
		5V			3		
I _{STB4}	Standby Current	3V	output pin floating, input pins at VDD, WDT enabled, LVR enabled		5		uA
		5V			5		
V _{IH1}	Input High Voltage	3V		1.5			V
		5V		2.5			
V _{IH2}	Input High Voltage	3V	P70, P71 input	1.2		3.0	V
		5V		2.0		3.0	
V _{IL1}	Input Low Voltage	3V				0.6	V
		5V				1.0	
V _{IHX}	Clock Input High Voltage	3V	OSCI		1.52		V
		5V			1.76		
V _{ILX}	Clock Input Low Voltage	3V	OSCI		0.76		V
		5V			1.0		
V _{OH1}	P5 Output High Voltage	3V	I _{OH} = 5mA		2.4		V
		5V			4.4		
V _{OH2}	P6 、 P7 Output High Voltage	3V	I _{OH} = 8mA		2.4		V
		5V			4.4		
V _{OL1}	Output Low Voltage	3V	I _{OL} = 10mA		0.6		V
		5V			0.25		
I _{PH}	Pull-high current	3V	Pull-high active, input pin at V _{SS}		30		uA
		5V			100		

八. 封装尺寸图

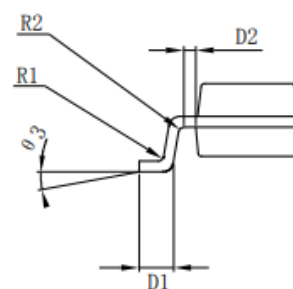
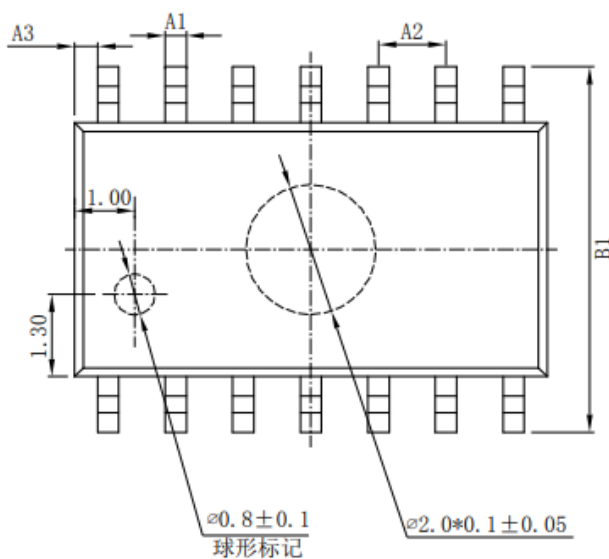
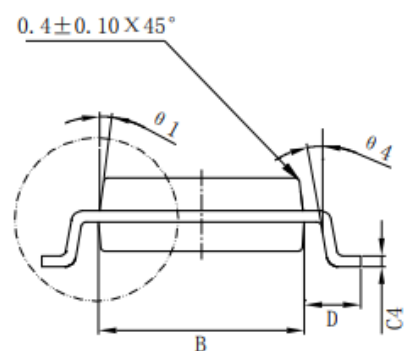
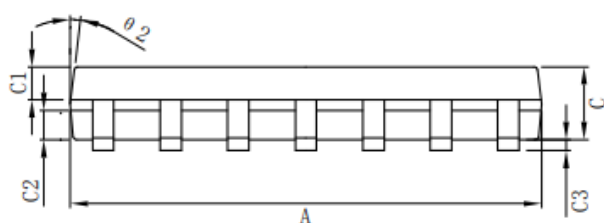
SOP24

标注 \ 尺寸	最小 (mm)	最大 (mm)	标注 \ 尺寸	最小 (mm)	最大 (mm)
A	15.19	15.39	C4	0.246	0.262
A1	0.406TYP		D	1.34	1.44
A2	1.22	1.32	D1	1.21TYP	
A3	0.457TYP		D2	0.18TYP	
B	7.40	7.60	R1	0.30TYP	
B1	10.206	10.406	R2	0.20TYP	
C	2.13	2.33	θ 1	12° TYP4	
C1	0.938	1.038	θ 2	12° TYP4	
C2	1.192	1.292	θ 3	0° ~ 8°	
C3	0.145	0.205			



SOP14

标注	尺寸	最小(mm)	最大(mm)	标注	尺寸	最小(mm)	最大(mm)
A		8.55	8.75	C4		0.193	0.213
A1		0.356	0.456	D		0.95	1.15
A2		1.27TYP		D1		0.40	0.70
A3		0.312TYP		D2		0.20TYP	
B		3.80	4.00	R1		0.20TYP	
B1		5.80	6.20	R2		0.20TYP	
C		1.40	1.60	θ 1		8° ~ 12° TYP4	
C1		0.60	0.70	θ 2		8° ~ 12° TYP4	
C2		0.55	0.65	θ 3		0° ~ 8°	
C3		0.05	0.25	θ 4		4° ~ 12°	



注意:

1. 以上信息如有更新，将不另做通知，请用户在使用前先确定手中的资料是否为最新版本。
2. 对于错误或不恰当操作所导致的后果，我们将不承担责任。

附：版本记录

版别	日期	更新内容	页次	备注
V1.0	2020-01-03	正式版本		

TX322 8-BIT MCU
