DATA SHEET

WL1109

8 位 CMOS OTP
单片机

用户参考手册

目 录

修正记录

日期	版本	内容	Page
2018/11/1	V001	初版	

WL1109

8 位 CMOS OTP 型单片机

1. 一般说明

WL1109是一个低成本、高性能、高效益的 CMOS OTP单片机。 它采用的 RISC 架构仅有单周期指。WL1109的性能大大高于同等价位的其它产品。8位宽的数据总线与12位宽的指令具有高度的对称性，与同类的单片机相比，大大提高其兼容性。

WL1109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。此系列单片机广泛被应在传感器信号处理、遥控器、消费性产品和子系统控制器等场合。

2. 特性

1. **512 x 12 bits OTP**程序存储器
2. **24 x 8 bits**通用寄存器
3. 工作电压范围
   - 1.8V ~ 3.6V @ 4MHz
4. 内建震荡器
   - BR4MHz：High accuracy 4MHz built-in OSC
     (± 1% @ V_{DD} = 1.8 ~ 3.6V, 0 ~ 40℃, after calibration)
     (± 1.5%@ V_{DD} = 1.8 ~ 3.6V, -10 ~ 50℃, after calibration)
     (± 2% @ V_{DD} = 1.8 ~ 3.6V, -25 ~ 75℃, after calibration)
5. **4**种时钟源除法器
   - Fcpu=Fosc/1
   - Fcpu =Fosc/2
   - Fcpu =Fosc/4
   - Fcpu =Fosc/8
6. **2**种处理器操作模式
   - 一般运作模式
   - 省电休眠模式(HALT mode)
7. 处理器核心
   - P10V3处理器核：36条强大指令
   - 一般指令：单周期指令
   - 跳转指令：双周期指令
8. 定时器**/**计数器
   定时器0, 定时器1：可读取的8 位预分频器的定时器
9. 输入输出端口
   - 内双向输入输出端口:
     PA4, PA5, PA6, PA7
   - 内置上拉电阻:
     PA4, PA5, PA6, PA7
   - 按键扫描唤醒功能:
     PA4, PA5, PA6, PA7
   - 下降沿触发唤醒功能:
     PA4, PA5, PA6, PA7
10. 硬件堆栈：4 级深度
11. 低功耗上电复位 (POR)
12. 低电复位(Low Voltage Reset LVR)
13. 振荡器起振定时 (OST)
14. 宽温度范围: -25 to 75 degree C
15. **I/O** 引脚的最大输出电流
    VDD=3.0V
    - 灌电流：-6mA (Voh = 2.7v)
    - 拉电流：10mA (Vol = 0.3v)

3. 引脚配置

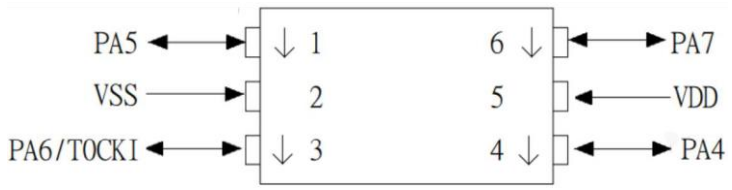
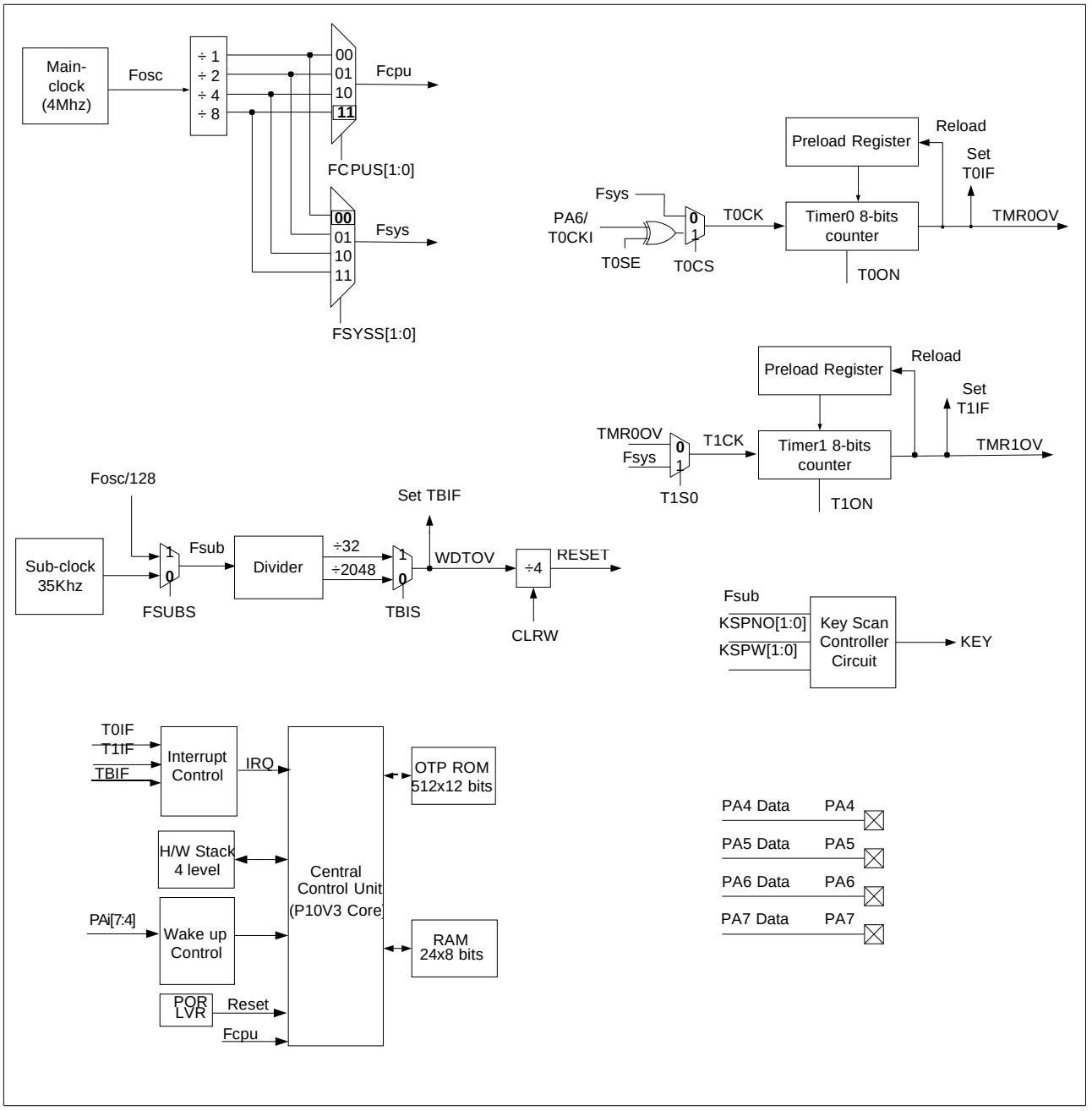


表 3-1：器件引脚说明 Pin Name	Pin Type	Buffer Type	Description
VDD	P	–	逻辑电路和 I/O 引脚的正向电源
VSS	P	–	逻辑电路和 I/O 引脚的接地参考点
PA4	I/O	ST	双向 I/O 引脚内置上拉电阻，具有唤醒功能 下降沿触发唤醒
PA5	I/O	ST	双向 I/O 引脚内置上拉电阻，具有唤醒功能 下降沿触发唤醒
PA6/T0CKI	I/O	ST	双向 I/O 引脚内置上拉电阻，具有唤醒功能 下降沿触发唤醒
PA7	I/O	ST	双向 I/O 引脚内置上拉电阻，具有唤醒功能 下降沿触发唤醒 漏极开路 NMOS 输出

图注：I = 输入, O=输出, P=电源, CMOS = CMOS输出, ST = 施密特触发器输入

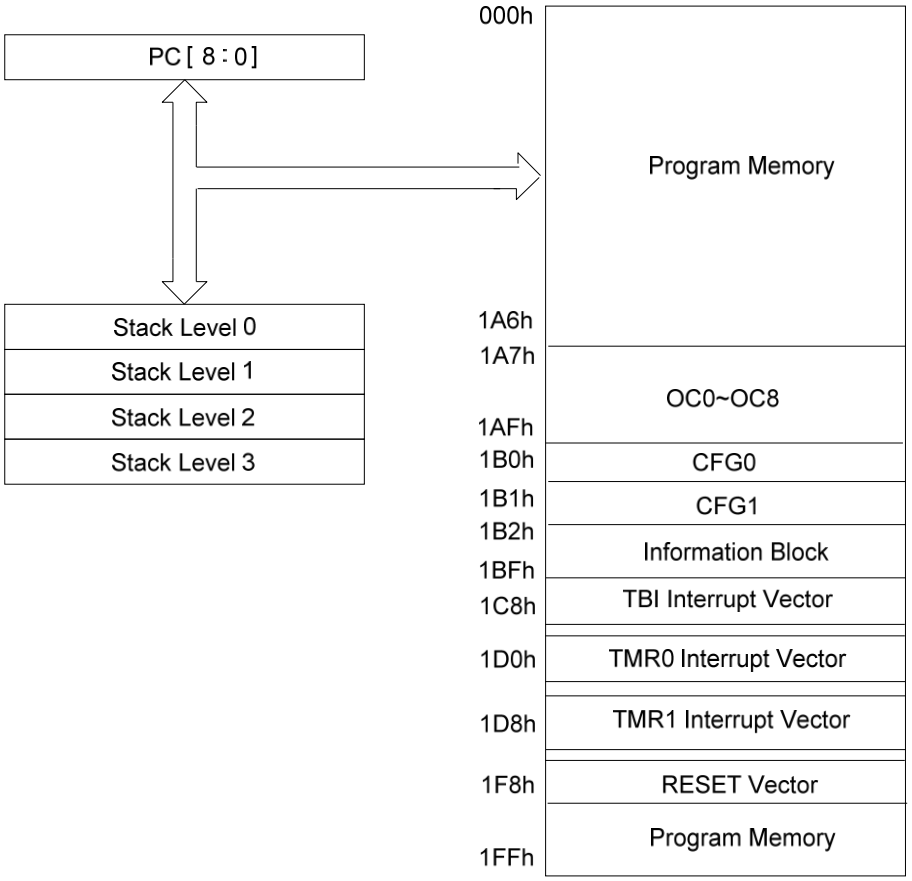
4. 框架图


Main-clock (4Mhz)
Fosc
÷ 1
÷ 2
÷ 4
÷ 8
00
01
10
11
Fcpu
FCPUS[1:0]
Fsys
FSYSS[1:0]
Preload Register
Reload
Set T0IF
Fsys
PA6/ T0CKI
T0SE
T0CS
T0CK
Timer0 8-bits counter
TMR0OV
T0ON
Preload Register
Reload
Set T1IF
TMR0OV
Fsys
T1S0
T1CK
Timer1 8-bits counter
TMR1OV
T1ON
Fosc/128
Sub-clock 35Khz
Fsub
FSUBS
Divider
÷32
÷2048
TBIS
Set TBIF
WDTOV
÷4
RESET
CLRW
Fsub
KSPNO[1:0]
KSPW[1:0]
Key Scan Controller Circuit
KEY
T0IF
T1IF
TBIF
Interrupt Control
IRQ
OTP ROM 512x12 bits
H/W Stack 4 level
Central Control Unit (P10V3 Core)
PAi[7:4]
Wake up Control
RAM 24x8 bits
POR LVR
Reset
Fcpu
PA4 Data
PA4
PA5 Data
PA5
PA6 Data
PA6
PA7 Data
PA7

5. 内存器

WL1109中的存储器包括程序存储器，配置存储器，数据存储器和控制寄存器。

5.1. 程序存储器



程序计数器 **(PC)**：WL1109 器件而言，具备一个9位程序计数器，可用来寻址 512 x 12 的程序存储器空间。在指令被执行后，程序计数器（PC）将包含将要执行的下一条指令的地址。PC 值在每个指令周期加一，除非该指令更改了PC值。但是，如果程序执行JSR 或JMP 指令时，PC 则指向特定地址。对于 JMP 和 JSR 指令来说，JMP 和 JSR 指令字将提供 PC 的值。PC 锁存器（PC Latch，PCL）将被映射到 PC<7:0>，注意只有低8位，即程序计数器低字节寄存器PCL，是可以让使用者直接读写的。然而 SJMP 和 SJSR 指令来说必须先行设定高位字节地址放在PCH<0>（参见表5-1）。

表**5-1**：装载 **PC** 的跳转指令说明

SJMP Instruction	PC = +/- 64 words
SJSR Instruction	PC = +/- 64 words
JMP Instruction	PC = Instruction word2[8:0]
JSR Instruction	PC = Instruction word2[8:0]
Relative Branch (modify PCL)	PC = +/- offset ({PCH[0], PCL} → PC)
Reset	PC = 1F8h
Interrupt	PC = 1C8h (TBI) PC = 1D0h (TMR0) PC = 1D8h (TMR1)
RET and RETI Instructions	PC = STACK[top]
Normal Instructions	PC = PC + 1

堆栈 **(STACK)**：WL1109 器件是 12 位宽，4 级深的硬件压入 / 弹出堆栈。堆栈存储器由12-bit寄存器和2-bit计数器组成。JSR 指令将把堆栈 1 的当前值压入第 2 级堆栈，然后将当前程序计数器的值加 1，并将其压入第 1 级堆栈，如果连续执行了4个以上的 JSR 指令，则将仅储存最后4次的返回地址。 RET 指令将第 1 级堆栈中的内容弹出到程序计数器， 然后将第 2 级堆栈中的内容复制到第 1 级堆栈中。 如果连续执行了4个以上的RET指令，则将用原来存储在第 4 级堆栈中的地址来填充堆栈。

5.2. 配置位存储器 (Configuration ROM)

WL1109 配置位由 12 位组成各有配置位存储器。可以编程配置字来选择不同的器件配置。(参见表 5-2)。在正常执行 过程中不能访问这些单元，但可在编程 / 校验模式中对它们进行读写。

表 **5-2**：配置位存储器

	11	10	9	8	7	6	5	4	3	2	1	0
1A7h	-	-	-	-	OC8							
1A8h	-	-	-	-	OC7							
1A9h	-	-	-	-	OC6							
1AAh	-	-	-	-	OC5							
1ABh	-	-	-	-	OC4							
1ACh	-	-	-	-	OC3							
1ADh	-	-	-	-	OC2							
1AEh	-	-	-	-	OC1							
1AFh	-	-	-	-	OC0							
CFG0	PROT	-	LVRC		DRT	WUP	-	-	-	-	-	-
CFG1	WOC				OCS							

PROT : 源码保护位
1 = 无，全部有点可读
0 = 全部，无可读性

LVRC[1:0]：LVR / LVD校准位。 LVR目标电压Vorg为1.65V。
VLVR 规格 是 1.65V +/- 0.1V。
00= Vorg + 0.1V 01= Vorg 10= Vorg - 0.1V 11= Vorg

WUP: HALT mode时看门狗定时器唤醒 CPU 使能位
1 = 看门狗定时器唤醒 CPU 时执行进入HALT mode时的下一个指令
0 = 看门狗定时器唤醒 CPU 时执行复位

DRT: 上电延时定时器
1 = 29ms
0 = 3.6ms

WOC :子时钟校准数据
1111 =最慢
0000 =最快

OCS: 主时钟校准数据地址选择

OCS	ADDRESS
11111111	1AFh
11111110	1AEh
11111100	1ADh
11111000	1ACh
11110000	1ABh
11100000	1AAh
11000000	1A9h
10000000	1A8h
OTHERS	1A7h

OC0~OC8: 主时钟校准数据
主时钟的输出频率与OC<7:0>成比例。调整一级为0.5%。
11111111 = 最慢
00000000 = 最快

5.3. 数据存储器

图 **5-2:**数据存储器和特殊功能寄存器框图

00h	INDX
01h	FSR
02h	Unimplemented
03h	Unimplemented
04h	STATUS
05h	PCL
06h	PCH
07h	ACC
08h	TBDL
09h	TBDH
0Ah	PIE
0Bh	PIF
0Ch	Unimplemented
0Dh	TMRC
0Eh	TBPL
0Fh	TBPH
10h	PAD
11h	PAR
12h	Unimplemented
13h	Unimplemented
14h	Unimplemented
15h	Unimplemented
16h	Unimplemented
17h	Unimplemented
18h	Unimplemented
19h	Unimplemented
1Ah	Unimplemented
1Bh	Unimplemented
1Ch	Unimplemented
1Dh	Unimplemented
1Eh	Unimplemented
1Fh	Unimplemented

5.4. 特殊功能寄存器

表 **5-3**：殊功能寄存器列表

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0	Value on POR,LVR	Value on other reset
00h	INDX	Address this location for indirect addressing								xxxx xxxx	uuuu uuuu
01h	FSR	-	-	b5	b4	b3	b2	b1	b0	--00 0000	--uu uuuu
04h	STATUS	-	-	-	TOB	PDB	Z	H	C	---1 1xxx	---p puuu
05h	PCL	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0	0000 0000	0000 0000
06h	PCH	-	-	-	-	-	-	-	PCH0	---- ---1	---- ---1
07h	ACC	b7	b6	b5	b4	b3	b2	b1	b0	xxxx xxxx	uuuu uuuu
08h	TBDL	b7	b6	b5	b4	b3	b2	b1	b0	xxxx xxxx	xxxx xxxx
09h	TBDH	-	-	-	b4	b3	b2	b1	b0	---- xxxx	---- xxxx
0Ah	PIE	GIE	-	-	TBIE	-	-	T1IE	T0IE	0--0 --00	0--0 --00
0Bh	PIF	-	-	-	TBIF	-	-	T1IF	T0IF	---0 --00	---0 --00
0Dh	TMRC	T0SE	T0CS	-	-	-	-	T1ON	T0ON	00-- --00	00-- --00
0Eh	TBPL	TBP7	TBP6	TBP5	TBP4	TBP3	TBP2	TBP1	TBP0	0000 0000	uuuu uuuu
0Fh	TBPH	TPINC	-	-	-	-	-	-	TBP8	0--- ---0	u--- ---u
10h	PAD	b7	b6	b5	b4	-	-	-	-	0000 ----	0000 ----
11h	PAR	b7	b6	b5	b4	-	-	-	-	xxxx ----	xxxx ----

图注：x =未知，u =不变，p =值取决于条件， - =未实现。

◆ **INDX (00h)**

INDX 寄存器不是实际存在的寄存器。寻址 INDX 寄存器会导致间接寻址，使用 INDX 寄存器可以实现间接寻址。任何使用 INDX 寄存器的指令实际上访问的是文件选择寄存器（File Select Register，FSR）指向的数据。间接读 INDX 本身会产生00h。而使用间接寻址对 INDX 寄存器进行写操作将导致执行一个空操作（虽然可能会影响状态位）。

◆ **FSR (01h)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
01h (r/w)	FSR	-	-	b5	b4	b3	b2	b1	b0

文件选择寄存器（File Select register）FSR 是一个6 位指针指向FSR 寄存器中的寄存器，这是间接寻址。

◆ **STATUS (04h)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
04h (r/w)	STATUS	-	-	-	TOB	PDB	Z	H	C

C: 进位/借位位(对于ADD和SUB以及RRC和RLC指令)

ADD	SUB	RRC 和 RLC
1 = 发生进位 0 = 未发生进位	1 = 未发生借位 0 = 发生借位	分别装载最低位或最高位

H: DC:半进位/借位位(ADD和SUB指令)

ADD	SUB
1 = 结果的第 4 个低位向高位发生了进位 0 = 结果的第 4 个低位未向高位发生进位	1 = 结果的第 4 个低位未向高位发生借位 0 = 结果的第 4 个低位向高位发生了借位

Z: 全零位

1 = 算术运算或逻辑运算结果为零

0 = 算术运算或逻辑运算结果不为零

PDB: 掉电位

(仅读) 1 = 上电或执行了 CLR WDT 指令

0 = 执行了 HALT 指令

TOB: 超时位

(仅读) 1 = 上电或执行了 CLR WDT 或 HALT 指令

0 = 发生了 WDT 超时

◆ **PCL (05h)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
05h (r/w)	PCL	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0

PC 锁存器(PC Latch, PCL)将被映射到 PC<7:0>。

◆ **PCH (06h)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
06h (r/w)	-	-	-	-	-	-	-	-	PCH0

PCH寄存器被使用在下列2种情况:

1.当写入到 PCL寄存器， 程序计数器（PC）会更新为PCH寄存器和PCL寄存器组合之地址
(PC<8> ← PCH<0>, PC<7:0]>← PCL<7:0>)

2. 当执行 MOVC 指令， PCH寄存器被用来当表格指针之高位
(TBDH, A ← ROM<PCH, A>)

◆ **ACC (07h)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
07h (r/w)	ACC	b7	b6	b5	b4	b3	b2	b1	b0

累加器，表格读取数据指令(MOVC)执行之后，表格数据高字节存储在TBDH中，表格数据低字节会被传送到ACC。

◆ **TBDL (08h)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
08h (r/w)	TBDL	b7	b6	b5	b4	b3	b2	b1	b0

执行PEEK TBR或POKE TBPL后， 程序存储器中表格数据的低字节，则被传送到TBDL。

◆ **TBDH (09h)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
09h (r/w)	TBDH	-	-	-	b4	b3	b2	b1	b0

表格读取数据指令(MOVC)执行或执行PEEK TBR或POKE TBPL之后，程序存储器中表格数据的高字节，则被传送到TBDH。

◆ **PIE(0Ah)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
0Ah(r/w)	PIE	GIE	-	-	TBIE	-	-	T1IE	T0IE

T0IE: Timer0 溢出中断使能位
1 = 使能 Timer0 溢出中断
0 = 禁止 Timer0 溢出中断

T1IE: Timer1 溢出中断使能位
1 = 使能 Timer1 溢出中断
0 = 禁止 Timer1 溢出中断

TBIE: Time-base Timer 定时中断使能位
1 = 使能中断
0 = 禁止中断

GIE: 全局中断使能位
1 = 使能所有未屏蔽的中断
0 = 禁止所有中断

◆ **PIF(0Bh)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
0Bh(r/w)	PIF	-	-	-	TBIF	-	-	T1IF	T0IF

T0IF: Timer0 溢出中断标志位
1 = Timer0 寄存器已经溢出（必须用软件清零）
0 = Timer0 寄存器没有溢出
T1IF: Timer1 溢出中断标志位
1 = Timer1 寄存器已经溢出（必须用软件清零）
0 = Timer1 寄存器没有溢出
TBIF: Time-base Timer 定时中断标志位
1 = 已经到时中断（必须用软件清零）
0 = 没有到时

◆ **TMRC (0Dh)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
0Dh(r/w)	TMRC	T0SE	T0CS	-	-	-	-	T1ON	T0ON

T0ON: Timer 0 ON / OFF控制位
0 = timer 0 off
1 = timer 0 on
T1ON: Timer 1 ON/OFF control bit
0 = timer 1 off
1 = timer 1 on
T0SE: Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚电平的下降沿递增
0 = 在 T0CKI 引脚电平的上升沿递增
T0CS: Timer0 时钟源选择位
1 = 选择 T0CKI 引脚上的传输信号作为时钟源
0 = 选择内部指令周期时钟

◆ **TBPL (0Eh)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
0Eh(r/w)	TBPL	TBP7	TBP6	TBP5	TBP4	TBP3	TBP2	TBP1	TBP0

TBP<7:0> 低位OTP ROM地址指针
POKE TBPL将执行两个操作：一个是更新TBPL，另一个是读取TBPH和TBPL寄存器指向的程序存储器。结果的高字节放在寄存器TBDH上，低字节放在寄存器TBDL上。POKE TBPL操作需要两个周期。

◆ **TBPH (0Fh)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
0Fh (r/w)	TBPH	TPINC	-	-	-	-	-	-	TBP8

TBP<8> : 高位OTP ROM地址指针
TPINC : TBP<8:0>增加1使能位
0= 关闭
1= 使能
执行PEEK TBR或POKE TBPL后，TBP<8:0>将增加1。

◆ **PAD (10h)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
10h (r/w)	PAD	b7	b6	b5	b4	-	-	-	-

PAD是PA端口数据寄存器。

◆ **PAR (11h)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
11h (r)	PAR	b7	b6	b5	b4	-	-	-	-

读PA端口数据，只可读。

图 **5-3**：控制寄存器框图

00h	OC
01h	OSCS
02h	Unimplemented
03h	WOC
04h	SP
05h	TBR
06h	SYSCLK
07h	Unimplemented
08h	Unimplemented
09h	Unimplemented
0Ah	Unimplemented
0Bh	TMR0
0Ch	TMR1
0Dh	Unimplemented
0Eh	Unimplemented
0Fh	TMCLK
10h	PAC
11h	PAW
12h	Unimplemented
13h	Unimplemented
14h	Unimplemented
15h	Unimplemented
16h	Unimplemented
17h	Unimplemented
18h	Unimplemented
19h	Unimplemented
1Ah	Unimplemented
1Bh	Unimplemented
1Ch	KSC
1Dh	Unimplemented
1Eh	Unimplemented
1Fh	Unimplemented

表 **5-4:** 控制寄存器列表

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0	Value on POR,LVR	Value on other reset
00h	OC	b7	b6	b5	b4	b3	b2	b1	b0	cccc cccc(1)	cccc cccc(1)
01h	OSCS	TBIS	-	-	-	-	-	-	FSUBS	0--- ----0	0--- ----0
03h	WOC	b7	b6	b5	b4	-	-	-	-	cccc ----(1)	cccc ----(1)
04h	SP	-	-	-	-	-	-	SP1	SP0	---- --11	---- --11
05h	TBR	-	-	-	-	-	-	-	-	---- ----	---- ----
06h	SYSCK	FSYSS[1:0]		-	-	-	-	FCPUS[1:0]		00-- --11	00-- --11
0Bh	TMR0	b7	b6	b5	b4	b3	b2	b1	b0	xxxx xxxx	uuuu uuuu
0Ch	TMR1	b7	b6	b5	b4	b3	b2	b1	b0	xxxx xxxx	uuuu uuuu
0Fh	TMCLK	-	-	-	-	-	T1S0	-	-	---- -0--	---- -0--
10h	PAC	b7	b6	b5	b4	-	-	-	-	1111 ----	1111 ----
11h	PAW	b7	b6	b5	b4	-	-	-	-	1111 ----	1111 ----
1Ch	KSC	KSEN	-	-	-	KSPW[1:0]		KSPNO[1:0]		0--- xxxx	0--- uuuu

图注：x =未知，u =不变，p =值取决于条件， - =未实现。

1:上电复位后配置位存储器的内容传送到寄存器(OC,WOC)

◆ **OC (00h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
00h (r/w)	OC	b7	b6	b5	b4	b3	b2	b1	b0

OC为内部OSC（内置RC4MHz）校准数据。 设置 “11111111” 导致内部OSC最慢，“00000000” 导致最快。

◆ **OSCS (01h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
01h (r/w)	OSCS	TBIS	-	-	-	-	-	-	FSUBS

TBIS: Time-base 中断源选择位
0 = Fsub/2048
1 = Fsub/32
FSUBS: Fsub 时钟源选择位
0 = Sub-clock
1 = Fosc/128
Note: 在休眠模式下， Fsub= Sub-clock。

◆ **WOC (03h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
03h (r/w)	WOC	b7	b6	b5	b4	-	-	-	-

WOC 为内部 OSC (Built-in RC35KHz) 校准数据。设置 “1111” 导致内部OSC最慢，“0000” 导致最快。

◆ **SP (04h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
04h (r)	SP	-	-	-	-	-	-	SP1	SP0

堆栈指针(仅读)是一个2位寄存器，在复位后初始化为03h。它在子程序调用或中断请求发生时递减，并在发生返回时递增。

◆ **TBR (05h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
05h (r/w)	TBR	-	-	-	-	-	-	-	-

PEEK TBR实际上读取TBPH和TBPL寄存器指向的程序存储器。 结果的高字节放在寄存器TBDH上，低字节放在寄存器TBDL上。 请注意，PEEK TBR操作需要两个周期。

◆ **SYSYCLK (06h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
06h (r/w)	SYSCK	FSYSS[1:0]		-	-	-	-	FCPUS[1:0]	

FCPUS[1:0]: Fcpu 分频比选择位
00 = Fosc/1　　01 = Fosc/2
10 = Fosc/4　　11 = Fosc/8
FSYSS[1:0]: Fsys 分频比选择位
00 = Fosc/1　　01 = Fosc/2
10 = Fosc/4　　11 = Fosc/8

◆ **TMR0 (0Bh) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
0Bh(r/w)	TMR0	b7	b6	b5	b4	b3	b2	b1	b0

TMR0是一个8位寄存器，用于设置定时器计数器0的重载值。

◆ **TMR1 (0Ch) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
0Ch(r/w)	TMR1	b7	b6	b5	b4	b3	b2	b1	b0

TMR1是一个8位寄存器，用于设置定时器计数器1的重载值。

◆ **TMCLK (0Fh) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
0Fh(r/w)	TMCLK	-	-	-	-	-	T1S0	-	-

T1S0: Timer 1 时钟源选择位
0 = TMR0OV
1 = Fsys

◆ **PAC (10h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
10h (r/w)	PAC	b7	b6	b5	b4	-	-	-	-

端口A输入/输出模式选择。“1”是输入模式，“0”是输出模式。

◆ **PAW (11h) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
11h (r/w)	PAW	b7	b6	b5	b4	-	-	-	-

端口A唤醒选择。'1'启用，'0'禁用。

◆ **KSC (1Ch) (**通过**PEEK / POKE**指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
1Ch(r/w)	KSC	KSEN	-	-	-	KSPW[1:0]		KSPNO[1:0]	

KSEN: 按键扫描开启使能位
0 = 按键扫描关闭
1 = 按键扫描开启
KSPNO[1:0]: 按键扫描脉冲非重迭选择位
00 = 1 个扫描脉冲宽度　　01 = 3 个扫描脉冲宽度
10 = 7 个扫描脉冲宽度　　11 = 保留
KSPW[1:0]: 按键扫描脉冲宽度选择位
00 = 2 个 Fsub 周期时间宽度　　01 = 4 个 Fsub 周期时间宽度
10 = 8 个 Fsub 周期时间宽度　　11 = 16 个 Fsub 周期时间宽度

6. 振荡器

6.1. 时钟系统框图

WL1109内置4MHz RC振荡器。复位完成后，振荡器起振定时器（OST），旨在确保芯片在晶振达到稳定之前始终处于 复位状态。就RC振荡器的工作和功能取决于许多变量来说，振荡器起振定时器就提供了8 个时钟周期的延迟，用来 确保器件在供电电压稳定之前处于复位状态（见表6-1）。另包含时钟除法器， 除率可选1, 2, 4, 8, 单片机频率为除 法器之输出。

图**6-1**：时钟系统框图

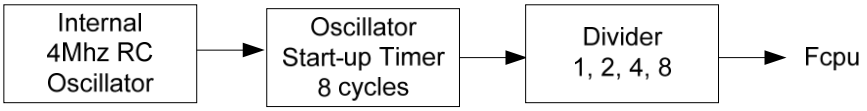


表**6-1**：**MCU**启动时间

模式	上电		从HALT唤醒
BR4M	DRT=0	DRT=1	8 Cycles
	3.6ms+8cycles	29ms+8 cycles	

Note: 1 cycle = 1/Fcpu

6.2. 内部振荡器 (BR4M)

WL1109集成了一个内部RC振荡器，在VDD = 3V和25°C时提供4 MHz输出时钟，校准后精度为±1%。
振荡器校准寄存器(OC)在上电后从Configuration1<7:0>获取校准数据的地址， 然后从地址获得校准数据。程序运行时，可以动态修改OC寄存器。内部OSC的输出频率与OC<7:0>的补码成正比，即OC<7:0> = 0xFF时内部OSC的频率最低，OC<7:0> = 0x00时的最高频率。OC的调整一级约为0.5%。

7. I/O 端口

7.1. 输入输出端口汇总

表 **7-1:** 输入输出端口汇总

	输入			输出		按键扫描
	上拉电阻	三态功能	唤醒功能	NMOS	CMOS	
PA4	V	V	V	V	V	V
PA5	V	V	V	V	V	V
PA6	V	V	V	V	V	V
PA7	V	V	V	V	-	V

7.2. I / O 端口寄存器

特殊功能寄存器

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
10h	PAD	b7	b6	b5	b4	-	-	-	-
11h	PAR	b7	b6	b5	b4	-	-	-	-

控制寄存器 **(**通过 **PEEK / POKE** 指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
10h	PAC	b7	b6	b5	b4	-	-	-	-
11h	PAW	b7	b6	b5	b4	-	-	-	-

表 **7-2:** 输入输出端口寄存器汇总

寄存器	寄存器读取时		寄存器写入		注
PAD	输入	寄存器数据	0: 设定输入端口为三态功能 1: 设定输入端口为上拉电阻		
	输出	寄存器数据	0: 设定输出端口为逻辑'0' 1: 设定输出端口为逻辑'1'		
PAC	寄存器数据		0: 寄存器第 n 位, 为输出端口 1: 寄存器第 n 位, 为输入端口		
PAW	寄存器数据		输入	0: 寄存器第 n 位,禁止唤醒功能 1: 寄存器第 n 位,使唤醒功能	
			输出	0: 寄存器第 n 位,CMOS 输出 1: 寄存器第 n 位,NMOS 输出	
PAR	PA 数据		无		

7.3 I / O端口电路图

图**7-1: PA7**结构图

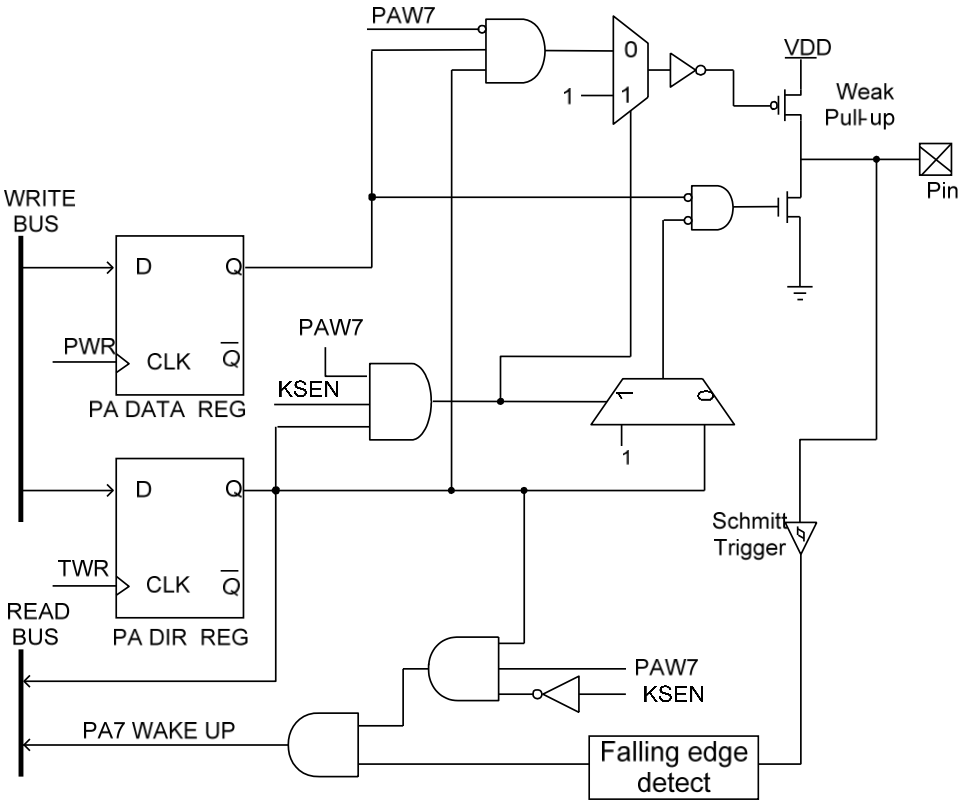


图**7-2: PA6**结构图

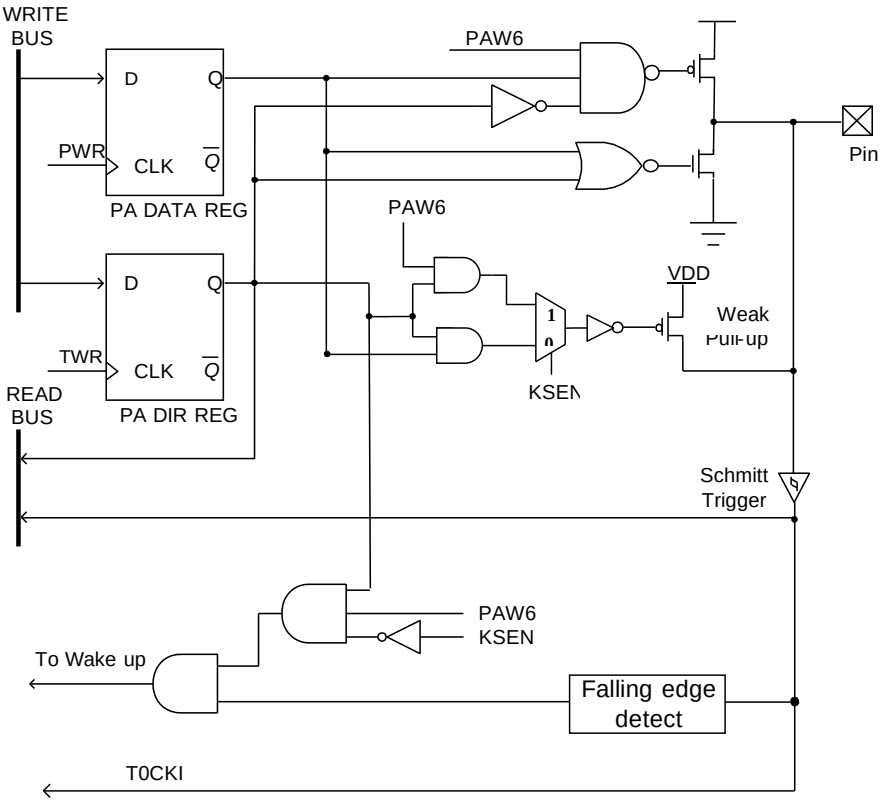
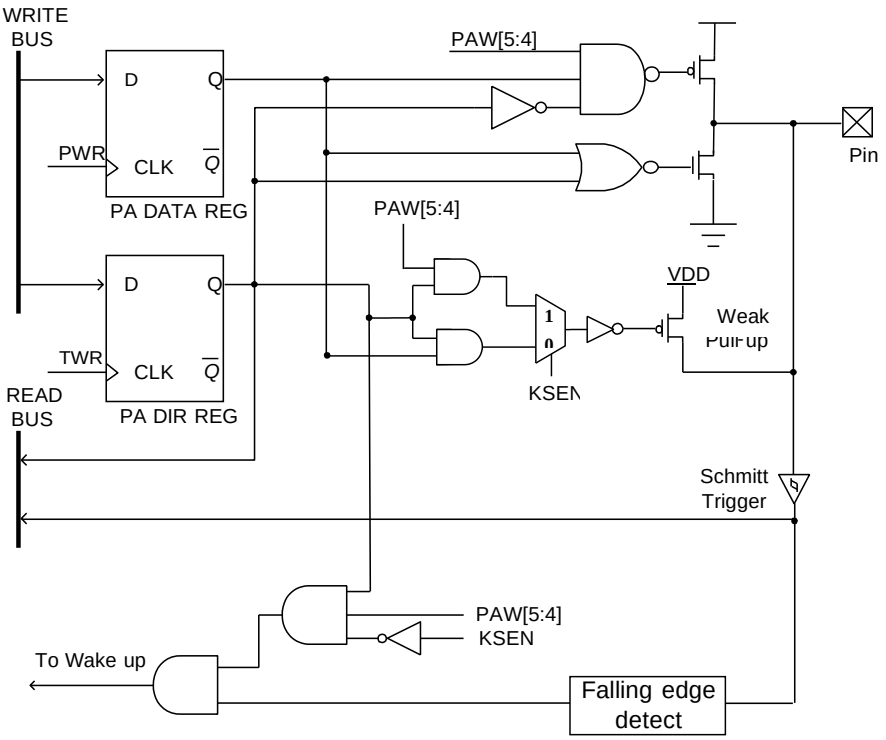


图**7-3: PA[5:4]** 结构图


WRITE
BUS
D
Q
PWR
CLK
Q
PA DATA REG
PAW[5:4]
Pin
PAW[5:4]
VDD
Weak
Pull-up
D
Q
TWR
CLK
Q
PA DIR REG
KSEN
READ
BUS
Schmitt
Trigger
PAW[5:4]
KSEN
To Wake up
Falling edge
detect

8. 单片机内建周边

8.1. Timer 0 & Timer 1 模块

Timer0/Timer1 模块定时器是 8 位上数计数器，当 Timer0/Timer1 寄存器定时器产生从 FFh 至 00h 的溢出时，产生 Timer0/Timer1 中断。此溢出将 T0IF/T1IF 标志位置 1， 同时从预存寄存器， 重新加载定时器初值。

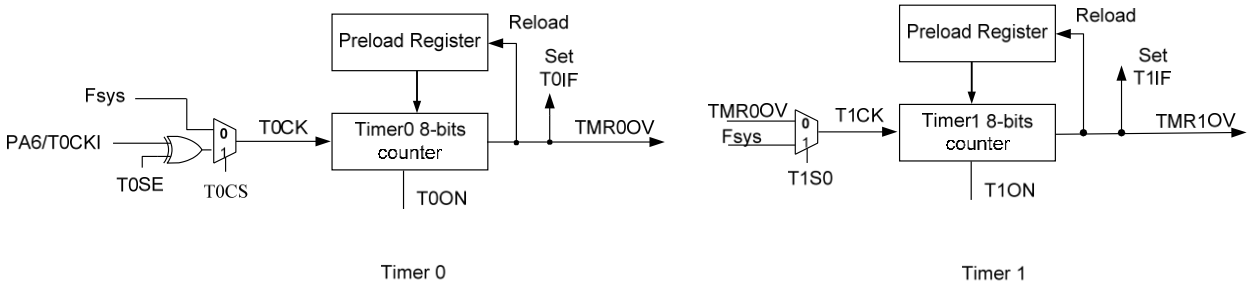
8.2. Timer 0 中断

当TMR0寄存器从FFh溢出到00h时，产生TMR0中断。该溢出将T0IF标志位置1。T0IE位清零可以屏蔽中断。在重新使能该中断之前，中断服务程序必须将T0IF位清零。选择T0CKI为钟源输入时，Timer0将根据T0SE的设置在T0CKI的每个上升沿或下降沿递增。

8.3. Timer 1 中断

当 TMR1 寄存器从 FFh 溢出到 00h 时，会产生 TMR1 中断。该溢出将 T1IF 标志位置 1。将 T0IE 位清零可以屏蔽中断。 在重新使能该中断之前，Timer1 模块中断服务程序用软件将 T1IF 位清零。由于定时器在休眠模式期间关闭，TMR1 中断无法将处理器从休眠状态唤醒。

图**8-1**：**Timer0**和**Timer1**框图



特殊功能寄存器

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
0Ah	PIE	GIE	-	-	TBIE	-	-	T1IE	T0IE
0Bh	PIF	-	-	-	TBIF	-	-	T1IF	T0IF
0Dh	TMRC	T0SE	T0CS	-	-	-	-	T1ON	T0ON

控制寄存器 **(**通过 **PEEK / POKE** 指令访问**)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
0Bh	TMR0	b7	b6	b5	b4	b3	b2	b1	b0
0Ch	TMR1	b7	b6	b5	b4	b3	b2	b1	b0
0Fh	TMCLK	-	-	-	-	-	T1S0	-	-

T0ON: Timer 0 ON / OFF控制位
0 = timer 0 off
1 = timer 0 on

T1ON: Timer 1 ON/OFF control bit
0 = timer 1 off
1 = timer 1 on

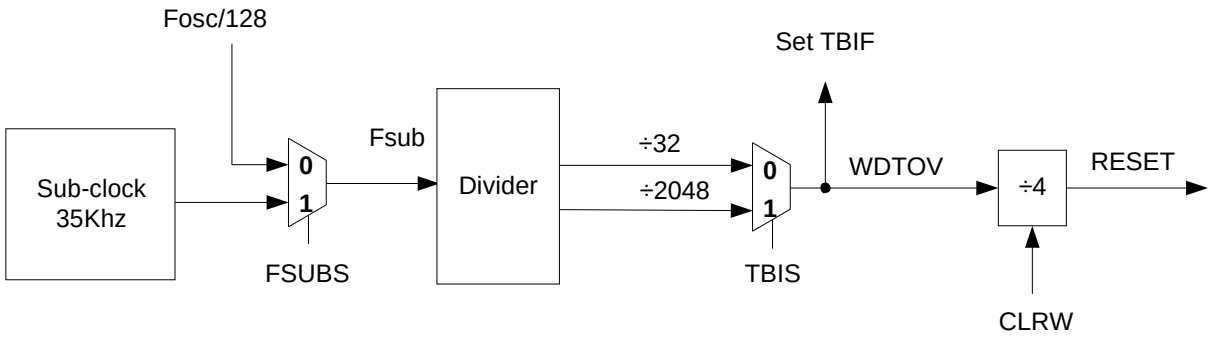
T0SE: Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚电平的下降沿递增
0 = 在 T0CKI 引脚电平的上升沿递增

T0CS: Timer0 时钟源选择位
1 = 选择 T0CKI 引脚上的传输信号作为时钟源
0 = 选择内部指令周期时钟

8.4. 看门狗定时器 & Time-Base Timer 模块

看门狗定时器（WDT）的架构如图 8-2。 WDT 的正常超时溢出周期为 0.9ms 或 58ms ，这个时间可由 TBIS 位 (OSCS<7>) 设置。当看门狗定时器超时溢出 4 次时会产生复位， TOB 位（STATUS<4>）将在看门狗定时器复位时清零。

图**8-2:** 看门狗定时器结构图

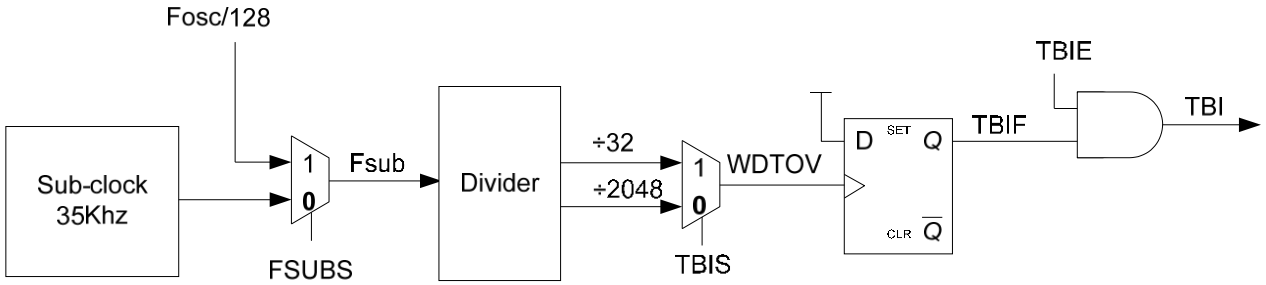


控制寄存器 **(**通过 **PEEK / POKE** 指令访问**)**

地址	名称	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
01h	OSCS	TBIS	-	-	-	-	-	-	FSUBS

Time-base Timer 模块如图 8-3， 当 Time-base Timer 溢出时，定时中断标志 TBIF 标志位置 1，且设置 PIE 寄存器 bit<4> TBIE 为 1 就会处于定时中断模式。每个定时中断事件发生时，将会进入中断服务程序。

图**8-3: Time-base Timer**结构图



8.5. 唤醒

执行 HALT 指令将进入掉电模式（休眠）
可以通过下列任一事件将器件从休眠状态唤醒：

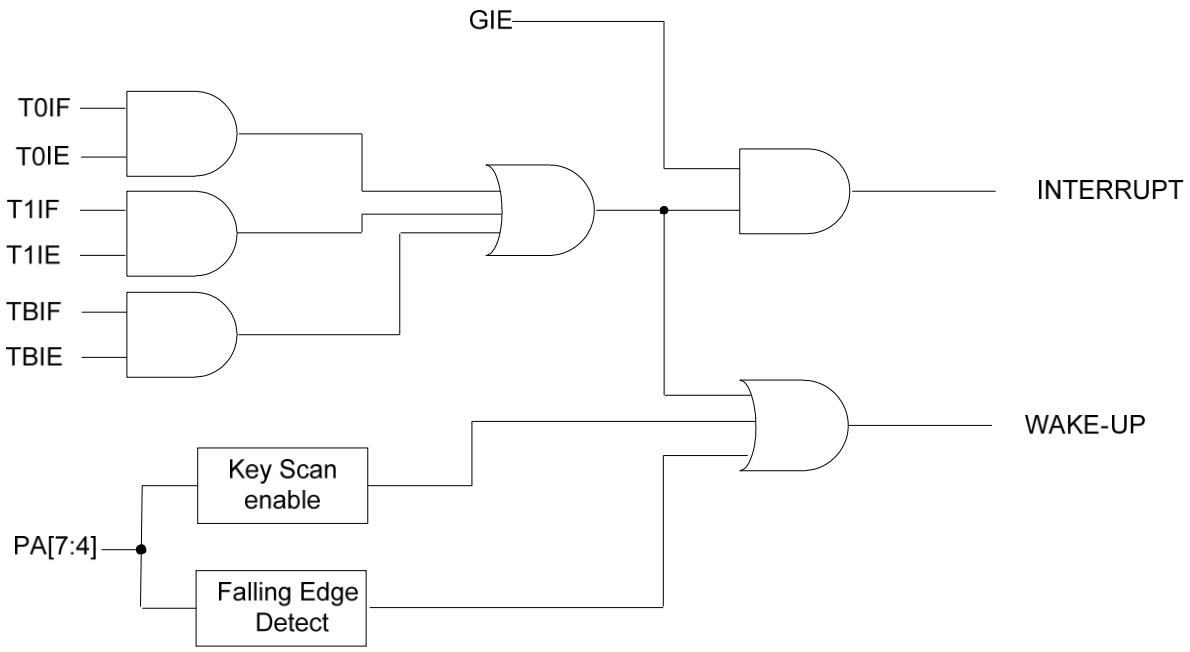
1. 看门狗定时器唤醒
2. TMR0 溢出中断、Time-Base Timer 定时中断唤醒
3. 按键扫描唤醒
4. PA[7:4] 下降沿触发唤醒

唤醒后可用状态寄存器中的TO 和PD 位判断器件复位的原因。PD 位在上电时置位，在进入休眠时清零。如果发生了WDT 唤醒将使TO 位清零。在执行HALT 指令时，下一个指令（PC+1）被预取。要通过中断事件唤醒器件，必须将相应的中断使能位置位（即允许中断）。唤醒与GIE 位的状态无关。如果GIE 位清零（禁止），器件仍继续执行HALT 指令以后的指令。 如果GIE 位被置位（使能），器件执行HALT 指令之后的指令，然后跳转到中断地址。 如果不想执行HALT 指令以后的指令，用户应该在HALT 指令后面放置一条NOP 指令。如果全局中断（GIE 清零）被禁止，但所有中断源都将其各自的中断使能位和相应的中断标志位置位，器件将立即从休眠状态唤醒。HALT 指令被完整执行。

表**8-1:** 唤醒设定条件表

唤醒源	设定条件	Note
Time-Base Timer 定时中断唤醒	TBIE=1	
TMR0 溢出中断唤醒	T0IE=1 和 T0CS=1	TIMER0使用外部时钟源:T0CKI
PA[7:4] 下降沿触发唤醒	PAW.n=1,PAC.n=1,PAD.n=0,KSEN=0	
按键扫描致能唤醒	PAW.n=1,PAC.n=1,PAD.n=0,KSEN=1	
看门狗定时器唤醒	上述唤醒源没被设定就会自动开启看门狗定时器唤醒	

图 **8.4 :** 中断和唤醒逻辑图



8.6. 按键扫描

WL1109 在休眠状态支持按键扫描功能。 请参考 图 8-5 和 图 8-6 可了解键扫描功能以及其应

特殊功能寄存器

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
10h	PAD	b7	b6	b5	b4	-	-	-	-

控制寄存器 **(Accessed by PEEK/POKE Instruction)**

地址	名称	B7	B6	B5	B4	B3	B2	B1	B0
10h	PAC	b7	b6	b5	b4	-	-	-	-
11h	PAW	b7	b6	b5	b4	-	-	-	-
1Ch	KSC	KSEN	-	-	-	KSPW[1:0]		KSPNO[1:0]	

KSEN: 按键扫描开启使能位
0 = 按键扫描关闭
1 = 按键扫描开启

KSPNO[1:0]: 按键扫描脉冲非重迭选择位
00 = 1 个扫描脉冲宽度　　01= 3 个扫描脉冲宽度
10 = 7 个扫描脉冲宽度　　11= 保留

KSPW[1:0]: 按键扫描脉冲宽度选择位
00 = 2 个 Fsub 周期时间宽度　　10 = 4 个 Fsub 周期时间宽度
10 = 8 个 Fsub 周期时间宽度　　11 = 16 个 Fsub 周期时间宽度

表 **8-2:** 按键扫描设置表

PAW.n	PAC.n	KSEN	PAD.n	PA Function		
				Wake up	Key scan	I/O
1	1	0	x	Wake up	-	input
1	1	1	x	-	Key scan	Key scan
0	0	0	x	-	-	NMOS output
1	0	0	x	-	-	CMOS output
x	1	0	0	-	-	Input tri-state
x	1	0	1	-	-	Input pull high

表 **8-3: Key scan time table: (Fix 16 scan lines)**

KSPW[1:0]		KSPNO[1:0]		Keyboard Scan Cycle Time
0	0	0	0	1.83ms (16 x 2x 2 x 28.6us)
		0	1	3.66ms (16 x 2 x 4 x 28.6us)
		1	0	7.32ms (16 x 2 x 8 x 28.6us)
0	1	0	0	3.66ms (16 x 4 x 2 x 28.6us)
		0	1	7.32ms (16 x 4 x 4 x 28.6us)
		1	0	14.64ms (16 x 4 x 8 x 28.6us)
1	0	0	0	7.32ms (16 x 8 x 2 x 28.6us)
		0	1	14.64ms (16 x 8 x 4 x 28.6us)
		1	0	29.28ms (16 x 8 x 8 x 28.6us)
1	1	0	0	14.64ms (16 x 16 x 2 x 28.6us)
		0	1	29.28ms (16 x 16 x 4 x 28.6us)
		1	0	58.56ms (16 x 16 x 8 x 28.6us)

Note:
The BR35K frequency is 35KHz +/- 30%.
28.6us is its center frequency cycle time.

按键扫描输出行序列：
当 KSEN = 1 时，按键扫描输出将遵循以下顺序：

图 **8-5:** 按键扫描时序图

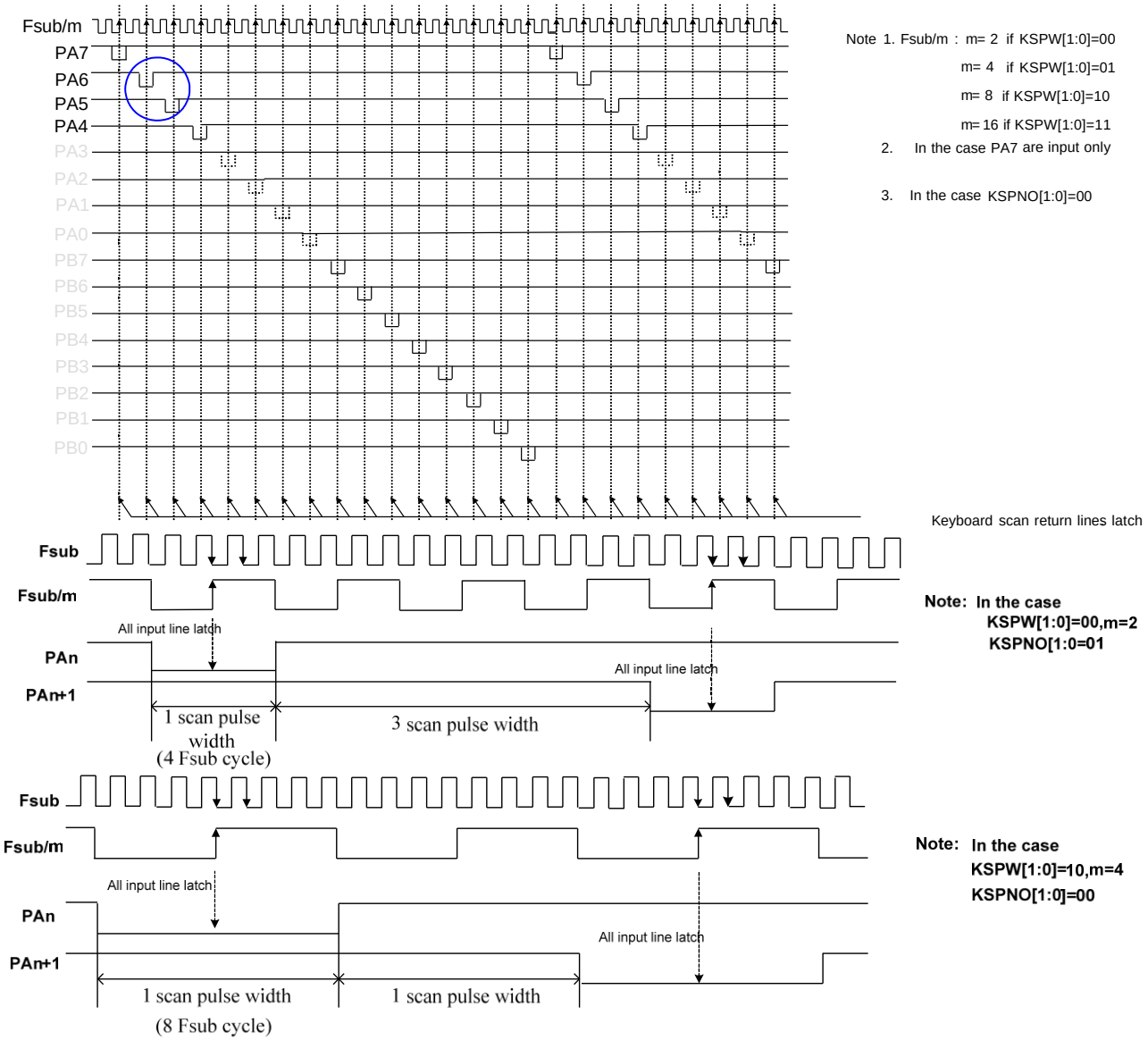
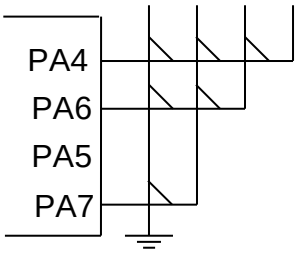


图 **8-6:** 按键扫描应用图



Note 1:
When PA7, PA6 and PA4 are key scan mode

Note 2:

In the case, PA7 is input only, PA5 used as normal I/O

9. OTP 烧录引脚

9.1. 编程引脚

表**9.1**串行模式编程信息

Programming information		
Writer connector		Parts: WL1109
Name	Description	Name
Power and clock Pins		
VDD	3V	VDD
GND	Ground	VSS
VPP	6.5V	PA7
Communication Pins		
Sync. CLK	Used to sync. data I/O	PA6
Serial Command/Data Bus	I/O pin	PA4

10. 指令集 (P10V3 CPU Core)

表**10-1:** 指令集

Arithmetic						
Instruction	r(Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
ADD A, r	0~3F		C, H , Z	1	1	r+A→ A
ADD r, A	0~3F		C, H, Z	1	1	r + A→ r
ADD A,#k		0~255	C, H, Z	1	1	k + A → A
DAA			C	1	1	A DAA → A
SUB A, r	0~3F		C, H, Z	1	1	r – A → A
SUB r, A	0~3F		C, H, Z	1	1	r – A → r
INC r	0~3F		Z	1	1	r +1 → r
Logic Operation						
Instruction	r(Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
ANL A, r	0~3F		Z	1	1	r & A→ A
CPL A, r	0~3F		Z	1	1	~r → A
ORL A, r	0~3F		Z	1	1	r | A→ A
XRL A, r	0~3F		Z	1	1	r ⊕ A → A
Rotate & Shift						
Instruction	r (Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
RRC r	0~3F		C	1	1	{C, r[7:1]} → r, r[0] → C
RLC r	0~3F		C	1	1	{r[6:0], C} → r, r[7] → C
Bit Operation						
Instruction	r(Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
RMBn r	0~3F			1	1	0 → r[n], n=0~7
SMBn r	0~3F			1	1	1 → r[n], n=0~7
Data Move						
Instruction	r(Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
MOV r, A	0~3F		(Z)	1	1	A → r If r=ACC, Z is updated.
MOV A, r	0~3F		Z	1	1	r → A
MOV A, #k		0~255		1		k → A
MOVC				2	1	ROM code (low byte) of {PCH[1:0], A} → A ROM code (high nibble) of {PCH[1:0], A} → TBDH
PEEK r	0~1F		Z	1	1	CR[r] → A
POKE r	0~1F			1	1	A → CR[r]
Branch						
Instruction	r(Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
DSZ r	0~3F			1 or 2(*)	1	r -1 → r, skip if 0
SJMP k		0~127		2	1	+/- 64 words → PC
SJSR k		0~127		2	1	PC + 1 → STACK[top], +/- 64 words → PC
JMP k		0~1023		2	2	k[9:0] → PC
JSR k		0~1023		2	2	PC + 1 → STACK[top], k[9:0] → PC
RET				2	1	STACK[top] → PC
RETI				2	1	1 →GIE, STA [CKtop] →PC
SBSn r	0~3F			1 or 2(*)	1	Skip if r[n] ≠ 0
SBRn r	0~3F			1 or 2(*)	1	Skip if r[n] = 0
SE r	0~3F			1 or 2(*)	1	Skip if A = r
Miscellaneous						
Instruction	r(Hex.)	#k(Dec.)	Flag	Cycles	Word	Description
CLR r	0~3F		Z	1	1	0→ r
CLRW			TOB, PDB	1	1	0 → DIV 4 CNT 1 → TOB, 1 → PDB
HALT			TOB, PDB	1	1	0 → DIV 4 CNT 1 → TOB, 0 → PDB
NOP				1	1	No operation
XCH r	0~3F			1	1	A ←→ r

(*) if skip then the next instruction is discarded, and the NOP is executed instead, making this a 2 cycle instruction

10.1. 指令描述

ADD	Add A and Reg
Format:	ADD r, A
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r + A -> r
Flag Affected:	C, H, Z
Description:	Add A with register r. Result is stored back in r.

ADD	Add A and Reg
Format:	ADD A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r + A -> A
Flag Affected:	C, H, Z
Description:	Add A with register r. Result is stored back in A.

ADD	Add A and literal
Format:	ADD A, #k
Cycles:	1
Operands:	0 ≤ k ≤ 255
Operations:	k + A -> A
Flag Affected:	C, H, Z
Description:	Add A with 8-bit literal ‘k’. Result is stored back in A.

ANL	Logic AND A with Reg
Format:	ANL A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r & A -> A
Flag Affected:	Z
Description:	Logic AND A with register r. Result is stored back in A.

CLRW	Clear watchdog timer
Format:	CLRW
Cycles:	1
Operands:	None
Operations:	0 -> DIV4 counter 1 -> TOB, 1 ->PDB
Flag Affected:	TOB, PDB
Description:	Clear divided by 4 counter. Flags TOB and PDB are set.

CLR	Clear register
Format:	CLR r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	0 -> r, 1 -> Z
Flag Affected:	Z
Description:	Register r is clear and Z is set.

CPL	Complement
Format:	CPL A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	~r -> A
Flag Affected:	Z
Description:	Complement r. Result is stored back in A.

DAA	Decimal-adjust after addition
Format:	DAA
Cycles:	1
Operands:	A
Operations:	If A[3:0] > 9 or H = 1 A[3:0] + 6 -> A[3:0], H1 =~H else A[3:0] -> A[3:0], H1 = 0 If A[7:4] + H1 > 9 or C = 1 A[7:4] + H1 + 6 -> A[7:4], C = 1 else A[7:4] + H1 -> A[7:4], C = C
Flag Affected:	C
Description:	Adjust data in register A from hexadecimal to decimal. Result is stored back in A.

DSZ	Decrement, skip if 0
Format:	DSZ r
Cycles:	1 (2)
Operands:	0 ≤ r ≤ 63
Operations:	r – 1 -> r, skip if 0
Flag Affected:	None
Description:	Decrement r. Result is stored back in r. If result is 0, skip next instruction by executing a NOP.

HALT	Standby
Format:	HALT
Cycles:	1
Operands:	None
Operations:	0 -> DIV4 counter 1 -> TOB, 0 -> PDB
Flag Affected:	TOB, PDB
Description:	Clear divided by 4 counter. Flag TOB is set and PDB is clear. Then the device is put into power-down mode and the oscillator stopped.

INC	Increment
Format:	INC r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r + 1 -> r
Flag Affected:	Z
Description:	Increment r. Result is stored back in r.

JMP	Unconditional branch
Format:	JMP k
Cycles:	2
Operands:	0 ≤ k ≤ 1023
Operations:	K[9:0] -> PC
Flag Affected:	None
Description:	JMP is an unconditional branch. The immediate address k[9:0] is loaded into PC.

JSR	Call subroutine
Format:	JSR k
Cycles:	2
Operands:	0 ≤ k ≤ 1023
Operations:	PC + 1 -> STACK[top], K[9:0] -> PC.
Flag Affected:	None
Description:	Call subroutine. Return address (PC+1) is pushed onto the STACK. The immediate address k[9:0] is loaded into PC.

MOV	Move between Register and A
Format:	MOV r, A
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	A -> r
Flag Affected:	(Z)
Description:	Move A to r. If r=ACC, Z flag is updated.

MOV	Move between Register and A
Format:	MOV A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r -> A Flag
Affected:	Z
Description:	Move r to A.

MOV	Move literal to A
Format:	MOV A, #k
Cycles:	1
Operands:	0 ≤ k ≤ 255
Operations:	k -> A
Flag Affected:	None
Description:	Move 8-bit literal ‘k’ to A.

MOVC	Move the ROM code to A and TBDH
Format:	MOVC
Cycles:	2
Operands:	None
Operations:	ROM code (low byte) of {PCH[1:0], A} -> A ROM code (high nibble) of {PCH[1:0], A} -> TBDH[3:0]
Flag Affected:	None
Description:	The low byte of ROM code addressed by A is moved to A and the high nibble is to TBDH.

NOP	No Operation
Format:	NOP
Cycles:	1
Operands:	None
Operations:	None
Flag Affected:	None
Description:	No operation.

ORL	Logic OR A with Reg
Format:	ORL A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r | A -> A
Flag Affected:	Z
Description:	Logic OR A with register r. Result is stored back in A.

ORL	Logic OR A with literal
Format:	ORL A, # k
Cycles:	1
Operands:	0 ≤ k ≤ 255
Operations:	k | A -> A
Flag Affected:	Z
Description:	Logic OR A with 8-bit literal ‘k’. Result is stored back in A.

PEEK	Read value from CM
Format:	PEEK r
Cycles:	1
Operands:	0 ≤ r ≤ 31
Operations:	CR[r] -> A
Flag Affected:	Z
Description:	Read value from control register and store back in A.

POKE	Write A to CM
Format:	POKE r
Cycles:	1
Operands:	0 ≤ r ≤ 31
Operations:	A -> CR[r]
Flag Affected:	None
Description:	Write A to control register.

RET	Return from subroutine
Format:	RET
Cycles:	2
Operands:	None
Operations:	STACK[top] -> PC
Flag Affected:	None
Description:	Return from subroutine. The top of the stack is popped and loaded into PC.

RETI	Return from interrupt
Format:	RETI
Cycles:	2
Operands:	None
Operations:	1 -> GIE, STACK[top] -> PC
Flag Affected:	None
Description:	Return from interrupt. Set GIE flag. The top of the stack is popped and loaded into PC.

RLC	Rotate left through carry
Format:	RLC r
Cycles:	1
Operands:	0 ≤ r ≤
Operations:	{r[6:0], C} -> r, r[7] -> C
Flag Affected:	C
Description:	Rotate register r one bit from right to left through carry flag. The LSB of r is replaced by C flag and the MSB value is moved to C flag. Result is stored back in r.

RMBn	Bit clear
Format:	RMBn r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	0 -> r[n]
Flag Affected:	None
Description:	Bit ‘n’ in register r is clear.

RRC	Rotate right through carry
Format:	RRC r, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	{C, r[7:1]} -> r, r[0] -> C
Flag Affected:	C
Description:	Rotate register r one bit from left to right through carry flag. The MSB of r is replaced by C flag and the LSB value is moved to C flag. Result is stored back in r.

SBRn	Bit test, skip if zero
Format:	SBRn r
Cycles:	1 (2)
Operands:	0 ≤ r ≤ 63
Operations:	Skip if r[n] = 0
Flag Affected:	None
Description:	If r[n] = 0, skip next instruction by executing a NOP.

SBSn	Bit test, skip if not zero
Format:	SBSn r
Cycles:	1 (2)
Operands:	0 ≤ r ≤ 63
Operations:	Skip if r[n] ≠ 0
Flag Affected:	None
Description:	If r[n] ≠ 0, skip next instruction by executing a NOP.

SE	Skip if equal
Format:	SE r
Cycles:	1 (2)
Operands:	0 ≤ r ≤ 63
Operations:	Skip if A = r
Flag Affected:	None
Description:	If A = r, skip next instruction by executing a NOP.

SJMP	Unconditional branch
Format:	SJMP k
Cycles:	2
Operands:	0 ≤ k ≤ 127
Operations:	+/- 64 words -> PC
Flag Affected:	None
Description:	SJMP is an unconditional branch. The new location is limited to PC +/- 64.

SJSR	Call subroutine
Format:	SJSR k
Cycles:	2
Operands:	0 ≤ k ≤ 127
Operations:	PC + 1 -> STACK[top], +/- 64 words -> PC.
Flag Affected:	None
Description:	Call subroutine. Return address (PC+1) is pushed onto the STACK. The new location is limited to PC +/- 64.

SMBn	Bit set
Format:	SMBn r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	1 -> r[n]
Flag Affected:	None
Description:	Bit ‘n’ in register r is set.

SUB	Subtract A from Reg
Format:	SUB A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r – A -> A
Flag Affected:	C, H, Z
Description:	Subtract A from register r. Result is stored back in A.

SUB	Subtract A from Reg
Format:	SUB r, A
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r – A -> r
Flag Affected:	C, H, Z
Description:	Subtract A from register r. Result is stored back in r.

XCH	Exchange between A and Reg
Format:	XCH r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	A <-> r
Flag Affected:	None
Description:	The contents of A and register r are exchanged.

XRL	Logic exclusive OR A with Reg
Format:	XRL A, r
Cycles:	1
Operands:	0 ≤ r ≤ 63
Operations:	r ⊕ A -> A
Flag Affected:	Z
Description:	Logic exclusive OR A with register r. Result is stored back in A.

11. 电气规范

绝对最大额定值

VDD 相对于VSS 的电压 ……..... -0.3V~3.6V
其它引脚相对于VSS 的电压 …… V_{SS}-0.3V~V_{DD}+0.3V
储存温度……..…….... -50℃~125℃
偏置电压下的环境温度……-25℃~75℃

11.1. 交流特性

符号	器件特性	VDD	条件	最小值	典型值	最大值	单位
Fosc	内部 4MHz 振荡频率容差	1.8V~3.6V	0℃~+40℃,,经校正后	-	1	-	±%
			-10℃~+50℃,,经校正后	-	1.5	-	±%
			-25℃~+75℃, 经校正后	-	2	-	±%
V_{POR}	**VDD** 起始电压确保能够产生上电复位信号	-	-	-	V_{SS}	-	V
S_{VDD}	**VDD** 上升速率确保能够产生上电复位信号	-	-	0.05*	-	-	V/ms
T_{LVR}	欠压复位脉冲宽度			100			ms

11.2. 直流特性

工作温度=25℃

符号	器件特性	V_{DD}	条件	最小值	典型值	最大值	单位
I_{DD1}	供电电流 1		BR4M (4MHz)				mA
		3V			1		mA
I_{STB}	基本断电电流		System halt, WDT disable				uA
		3V				1	uA
I_{OH}	I/O Port 输出拉电流 (PA)						mA
		3V	V_{OH}=2.7V		-6		mA
I_{OL}	I/O Port 输出灌电流 (PA)						mA
		3V	V_{OL}=0.3V		10		mA
R_{PH}	PA 上拉电阻		-				KΩ
		3V	-		150		KΩ
V_{IL1}	输入低电压 1	-	-	0	-	$0.3V_{DD}$	V
V_{IH1}	输入高电压 1	-	-	$0.7V_{DD}$	-	V_{DD}	V
V_{IL2}	输入低电压对 T0CKI	-	-	0	-	$0.2V_{DD}$	V
V_{IH2}	输入高电压对 T0CKI	-	-	$0.8V_{DD}$	-	V_{DD}	V
LVR	欠压电压	-	-	-	1.75	-	V

11.3. BR4M 振荡器特性曲线图（典型值）

Test Condition: Capacitance of VDD to VSS need connect to 4.7uF at least.