芯睿科技股份有限公司 mikkon technology limited	总页数	133
	Doc No	MK9A50P
	版本	27

文件名称	MK9A50P 8 位单片机使用说明书

版次	生效日	ECN No.	制修订者	修订内容概要
21	2009.6.12		李崑旭	新颁。
22	2009.9.10		李崑旭 (Jemmy)	P124 $68~69 仅做为工作 RAM：删除 P26 & P96 PAD_CTL5：修改
23	2009.9.14		Jemmy	P55 PWM_OS=0，初始输出阶段是 H，当定时器溢出将变为 H è 初始输出阶段是 L，当定时器溢出将变为 H
24	2009.10.14		Jemmy	P118 ELON 1：ELP/ELC 开启 0：ELP/ELC 关闭，输出信号为低 P5，P28 & P118 内置低电池检测 – 2.56V，2.40V & 2.68V P26 & P96 PAD_CTL5 Bit4 – SEG23/PC[6] 控制 Bit3 – SEG24/PC[7] 控制 P51 TM0_LA ***（Data =/= FFh）***
25	2009.11.3		Jemmy	P91 PD_PUD bit4 是无效下拉控制
26	2009.12.2		Jemmy	P25 & P90 PC_CTL BIT4~3 è BIT7~0
27	2010.3.22		Jemmy	P11~14 SEG24/PC[7]，SEG23/PC[6] P98~111 & P27 modify key STROBE

MK9A50P

（低功耗 8 位单片机）

使 用 说 明 书

目 录

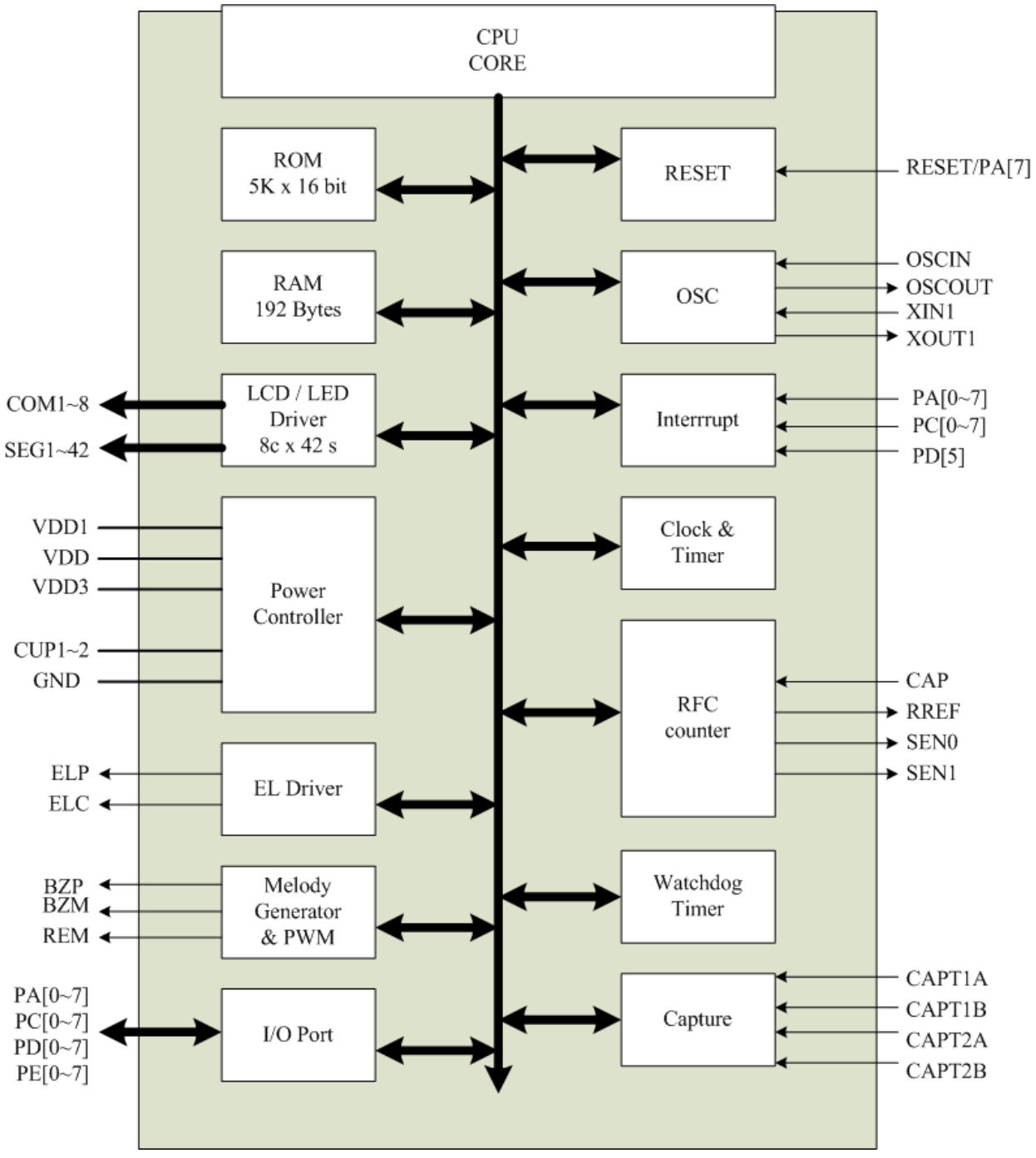
1. 概述

1.1 功能

- ROM 大小：5K * 16 bits
- RAM 大小：192 * 8 bits
- 堆栈：8 层
- LCD/LED 驱动器：**8com * 42seg**（**LDC RAM：8 x 42**）
  - 占空比：1/4、1/5、1/6、1/7、1/8 通过寄存器可选
  - 1/3 偏置电荷泵（LI 模式），3 种 LED 模式。
  - COM5~10 可选择做为 I/O 口
- I/O 口：31 个双向 I/O 口，1 个输入口
  - PA[0~6]可设置为上拉、下拉、正常输出、pmos 开漏或 nmos 开漏
  - PA7 仅输入带下拉
  - PC[0~7]可设置为下拉、正常输出或 pmos 开漏
  - PD[0~7]可设置为下拉、正常输出或 pmos 开漏
  - PE[0~7]可设置为下拉、正常输出或 pmos 开漏
- 脚位沿中断
  - 通用脚位：PA[0~7] & PC[0~7]
  - 单个脚位：PD[5]
- 按键选通功能 – 仅正常模式使用
  - 轮询模式：PA[0~6]、PC[0~7] & PD[3~4]
- 系统时钟：双时钟操作
  - 低速 -> 外部 32Khz 晶体、外部 R 振荡器或内部低速 RC 振荡器
  - 高速 -> 通过配置选项选择外部 4MHz 晶体、外部 R 振荡器或（700KHz 或 1.5MHz）内部高速 RC 振荡器
- 定时器 0 (TM0)：
  - 一个 8 位通用定时器
  - 远程输出（包括 REM 载体）
  - TM2 & 3 RFC 定时器基本输入
- 定时器 2 & 定时器 3（TM2 & TM3）：
  - 两个 8 位定时器：TM0，TM2 & TM3
  - 一个 16 位定时器：TM2 + TM3
  - 两个 8 位捕捉/RFC：TM2 & TM3
  - 一个 16 位捕捉/RFC：TM2 + TM3
  - 三个 8 位 PWM：TM2，TM3，（TM2 + TM3）

- 其他定时器基本源
  - PH_IRQ
  - PH_CLK
  - 2HZ
  - 一个 16 位预分配器
  - 看门狗定时器
- 看门狗定时器 & 4’按键复位功能
  - CONFIG WDTE=0：4’按键复位使能 & 看门狗定时器禁止
  - CONFIG WDTE=1：4’按键复位禁止 & 看门狗定时器使能
- 内置一 RFC 通道 -- CAP、REF、SEN0 & SEN1
- 内置两种 PWM 输出 – PWM2 & PWM3
- 内置四个捕捉通道 – CAPT1A、CAPT1B、CAPT2A & CAPT2B
- 特殊定时器
  - 一个 16 位预分配器
  - 看门狗定时器
- IRQ 源：8
- 内置 EL 驱动电路
- 内置可编程 FREQ，Tone & REM 输出
- 多功能（BZ、BZM）输出 – Tone、FREQ、1Khz、2Khz、4Khz、PWM2 或 PWM3 输出
- 内置低电压复位（LVR）-1.5V，1.7V & 2.0V
- 内置低电池检测 – 2.56V，2.40V & 2.68V
- REM – 远程输出
- HALT 及 SLEEP 操作模式
- 快速指令周期时间：61us@32KHz 操作
- 低功率消耗（XIN1，XOUT1）：2uA（@ 32KHz 暂停模式、LCD 打开、无负载）

1.2 结构示意图

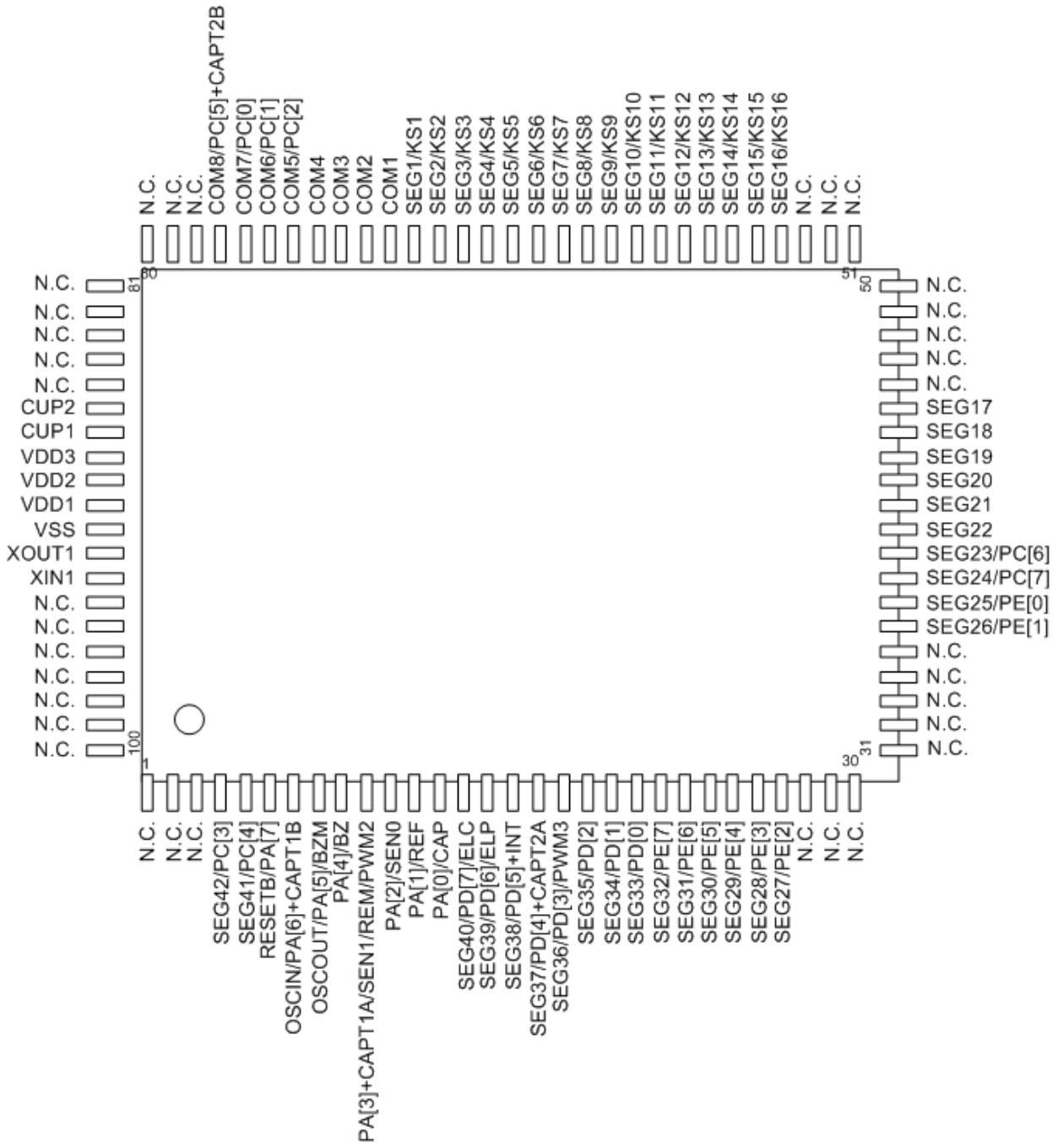

CPU
CORE
ROM
5K x 16 bit
RAM
192 Bytes
LCD / LED
Driver
8c x 42 s
Power
Controller
EL Driver
Melody
Generator
& PWM
I/O Port
RESET
OSC
Interrrupt
Clock &
Timer
RFC
counter
Watchdog
Timer
Capture
COM1~8
SEG1~42
VDD1
VDD
VDD3
CUP1~2
GND
ELP
ELC
BZP
BZM
REM
PA[0~7]
PC[0~7]
PD[0~7]
PE[0~7]
RESET/PA[7]
OSCIN
OSCOUT
XIN1
XOUT1
PA[0~7]
PC[0~7]
PD[5]
CAP
RREF
SEN0
SEN1
CAPT1A
CAPT1B
CAPT2A
CAPT2B

1.3 脚位分配

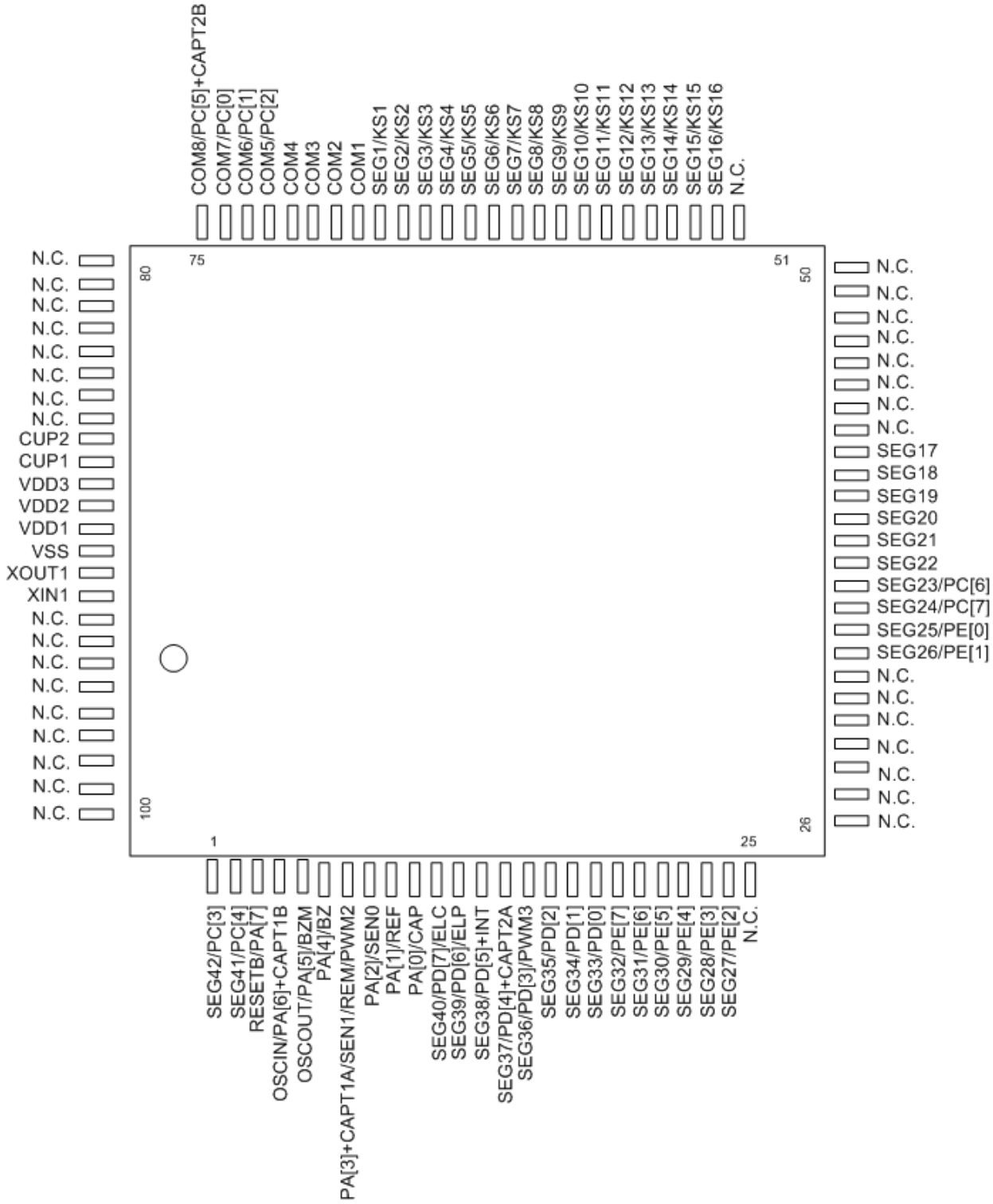
（COB 脚位分配）

No	Name	No	Name
1	SEG42/PC[3]	35	SEG16 /KS16
2	SEG41/PC[4]	36	SEG15 /KS15
3	RESETB /PA[7]	37	SEG14 /KS14
4	OSCIN /PA[6]+CAPT1B	38	SEG13 /KS13
5	OSCOUT /PA[5]/BZM	39	SEG12 /KS12
6	PA[4 /BZ	40	SEG11 /KS11
7	PA[3+CAPT1A/SEN1/REM/PWM2	41	SEG10 /KS10
8	PA[2] /SEN0	42	SEG9 /KS9
9	PA[1] /REF	43	SEG8 /KS8
10	PA[0] /CAP	44	SEG7 /KS7
11	SEG40 /PD[7] /ELC	45	SEG6 /KS6
12	SEG39 /PD[6] /ELP	46	SEG5 /KS5
13	SEG38 /PD[5]+INT	47	SEG4 /KS4
14	SEG37 /PD[4] +CAPT2A	48	SEG3 /KS3
15	SEG36 /PD[3] / PWM3	49	SEG2 /KS2
16	SEG35 /PD[2]	50	SEG1 /KS1
17	SEG34 /PD[1]	51	COM1
18	SEG33 /PD[0]	52	COM2
19	SEG32 /PE[7]	53	COM3
20	SEG31 /PE[6]	54	COM4
21	SEG30 /PE[5]	55	COM5 /PC[2]
22	SEG29 /PE[4]	56	COM6 /PC[1]
23	SEG28 /PE[3]	57	COM7 /PC[0]
24	SEG27 /PE[2]	58	COM8/PC[5] +CAPT2B
25	SEG26 /PE[0]	59	CUP2
26	SEG25 /PE[0]	60	CUP1
27	SEG24 /PC[7]	61	VDD3
28	SEG23 /PC[6]	62	VDD2
29	SEG22	63	VDD1
30	SEG21	64	GND
31	SEG20	65	XOUT1
32	SEG19	66	XIN1
33	SEG18	67	N.C.
34	SEG17		

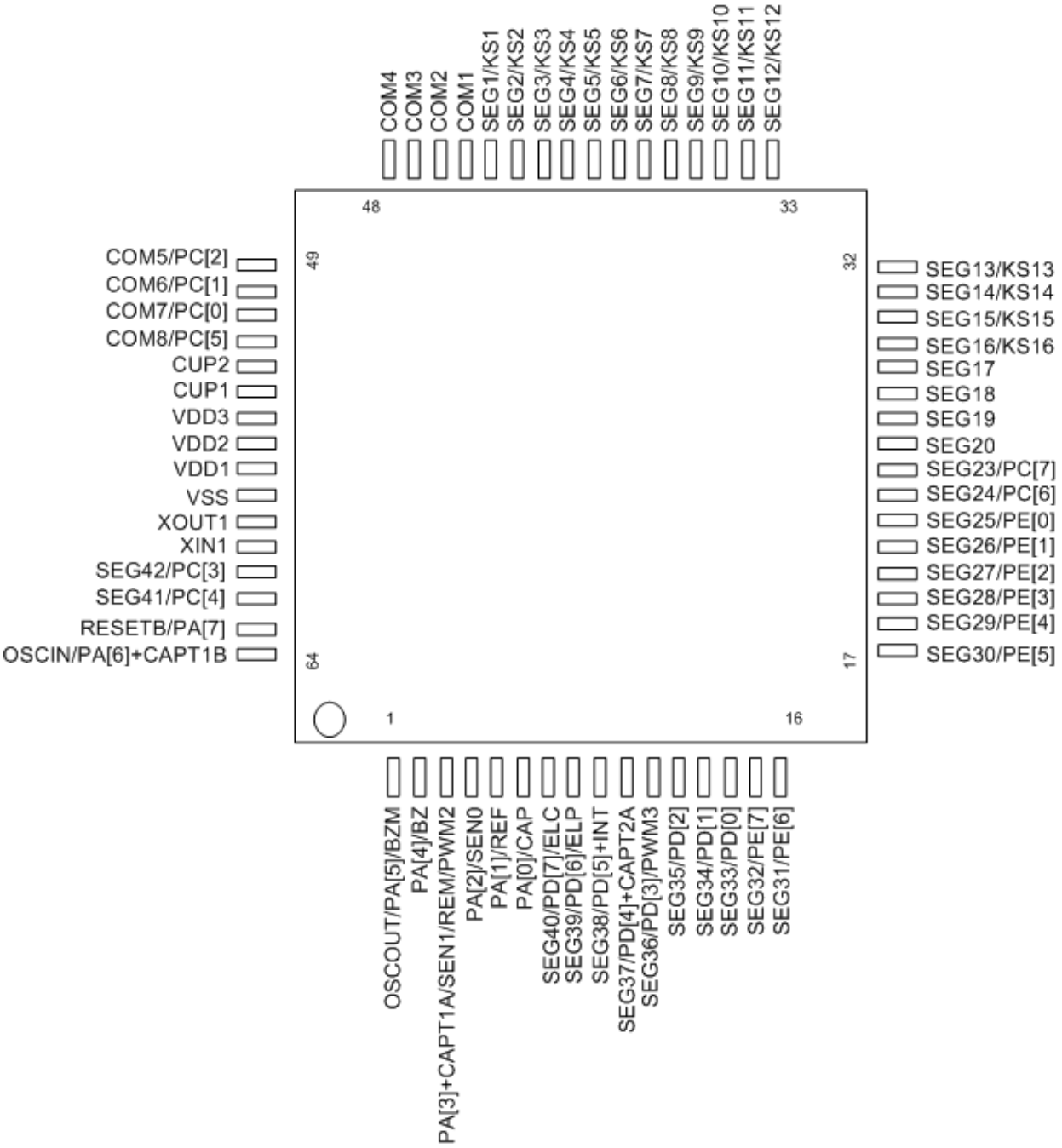
（QFP 100-PIN）


N.C.
N.C.
N.C.
COM8/PC[5]+CAPT2B
COM7/PC[0]
COM6/PC[1]
COM5/PC[2]
COM4
COM3
COM2
COM1
SEG1/KS1
SEG2/KS2
SEG3/KS3
SEG4/KS4
SEG5/KS5
SEG6/KS6
SEG7/KS7
SEG8/KS8
SEG9/KS9
SEG10/KS10
SEG11/KS11
SEG12/KS12
SEG13/KS13
SEG14/KS14
SEG15/KS15
SEG16/KS16
N.C.
N.C.
N.C.
81
80
51
50
N.C.
N.C.
N.C.
N.C.
N.C.
CUP2
CUP1
VDD3
VDD2
VDD1
VSS
XOUT1
XIN1
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
SEG17
SEG18
SEG19
SEG20
SEG21
SEG22
SEG23/PC[6]
SEG24/PC[7]
SEG25/PE[0]
SEG26/PE[1]
N.C.
N.C.
N.C.
N.C.
N.C.
100
1
30
31
N.C.
N.C.
N.C.
SEG42/PC[3]
SEG41/PC[4]
RESETB/PA[7]
OSCIN/PA[6]+CAPT1B
OSCOUT/PA[5]/BZM
PA[4]/BZ
PA[3]+CAPT1A/SEN1/REM/PWM2
PA[2]/SEN0
PA[1]/REF
PA[0]/CAP
SEG40/PD[7]/ELC
SEG39/PD[6]/ELP
SEG38/PD[5]+INT
SEG37/PD[4]+CAPT2A
SEG36/PD[3]/PWM3
SEG35/PD[2]
SEG34/PD[1]
SEG33/PD[0]
SEG32/PE[7]
SEG31/PE[6]
SEG30/PE[5]
SEG29/PE[4]
SEG28/PE[3]
SEG27/PE[2]
N.C.
N.C.
N.C.

（LQFP 100-PIN）


COM8/PC[5]+CAPT2B
COM7/PC[0]
COM6/PC[1]
COM5/PC[2]
COM4
COM3
COM2
COM1
SEG1/KS1
SEG2/KS2
SEG3/KS3
SEG4/KS4
SEG5/KS5
SEG6/KS6
SEG7/KS7
SEG8/KS8
SEG9/KS9
SEG10/KS10
SEG11/KS11
SEG12/KS12
SEG13/KS13
SEG14/KS14
SEG15/KS15
SEG16/KS16
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
CUP2
CUP1
VDD3
VDD2
VDD1
VSS
XOUT1
XIN1
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
N.C.
SEG17
SEG18
SEG19
SEG20
SEG21
SEG22
SEG23/PC[6]
SEG24/PC[7]
SEG25/PE[0]
SEG26/PE[1]
SEG42/PC[3]
SEG41/PC[4]
RESETB/PA[7]
OSCIN/PA[6]+CAPT1B
OSCOUT/PA[5]/BZM
PA[4]/BZ
PA[3]+CAPT1A/SEN1/REM/PWM2
PA[2]/SEN0
PA[1]/REF
PA[0]/CAP
SEG40/PD[7]/ELC
SEG39/PD[6]/ELP
SEG38/PD[5]+INT
SEG37/PD[4]+CAPT2A
SEG36/PD[3]/PWM3
SEG35/PD[2]
SEG34/PD[1]
SEG33/PD[0]
SEG32/PE[7]
SEG31/PE[6]
SEG30/PE[5]
SEG29/PE[4]
SEG28/PE[3]
SEG27/PE[2]
N.C.
1
25
26
50
51
75
80
100

（LQFP 64-PIN）


COM4
COM3
COM2
COM1
SEG1/KS1
SEG2/KS2
SEG3/KS3
SEG4/KS4
SEG5/KS5
SEG6/KS6
SEG7/KS7
SEG8/KS8
SEG9/KS9
SEG10/KS10
SEG11/KS11
SEG12/KS12
48
33
49
32
64
17
1
16
COM5/PC[2]
COM6/PC[1]
COM7/PC[0]
COM8/PC[5]
CUP2
CUP1
VDD3
VDD2
VDD1
VSS
XOUT1
XIN1
SEG42/PC[3]
SEG41/PC[4]
RESETB/PA[7]
OSCIN/PA[6]+CAPT1B
SEG13/KS13
SEG14/KS14
SEG15/KS15
SEG16/KS16
SEG17
SEG18
SEG19
SEG20
SEG23/PC[7]
SEG24/PC[6]
SEG25/PE[0]
SEG26/PE[1]
SEG27/PE[2]
SEG28/PE[3]
SEG29/PE[4]
SEG30/PE[5]
OSCOUT/PA[5]/BZM
PA[4]/BZ
PA[3]+CAPT1A/SEN1/REM/PWM2
PA[2]/SEN0
PA[1]/REF
PA[0]/CAP
SEG40/PD[7]/ELC
SEG39/PD[6]/ELP
SEG38/PD[5]+INT
SEG37/PD[4]+CAPT2A
SEG36/PD[3]/PWM3
SEG35/PD[2]
SEG34/PD[1]
SEG33/PD[0]
SEG32/PE[7]
SEG31/PE[6]

1.4 脚位描述

脚位名称	I/O	描　述
PA[0] /CAP	I,I/O	1. 按键输入脚位 2. 输入口带上拉，下拉或脚位唤醒 3. 输出口（正常输出，P开漏或N开漏） 4. CAP输入
PA[1] /REF	I,I/O	1. 输入口带上拉，下拉或脚位唤醒 2. 输出口（正常输出，P开漏或N开漏） 3. REF输出
PA[2] /SEN0	I,I/O	1. 输入口带上拉，下拉或脚位唤醒 2. 输出口（正常输出，P开漏或N开漏） 3. SEN0输出
PA[3]+CAPT1A/SEN1/ REM /PWM2	I,I/O	1. 输入口带上拉，下拉或脚位唤醒 2. 输出口（正常输出，P开漏或N开漏） 3. CAPT1A输入 4. REM输出 5. PWM2（TM2）输出 6. SEN1输出
PA[4] /BZ	I,I/O	1. 输入口带上拉，下拉或脚位唤醒 2. 输出口（正常输出，P开漏或N开漏） 3. BZM输出
OSCOUT/ PA[5]+BZM	I,I/O	1. 晶体输出 2. I/O口（上拉，下拉，开漏，脚位唤醒） 3. BZM输出
OSCIN /PA[6]+CAP1B	O,I/O	1. 晶体输入 2. 外部RC输入 3. I/O口（上拉，下拉，开漏，脚位唤醒） 4. CAP1B输入
RESETB/PA[7]/INT	I,I	1. 系统复位输入（浮动） 2. 输入口带下拉 3. 脚位变化时唤醒 4. 脚位中断
SEG[41] /PC[4]	I,O	1. 输入口带下位 2. 输出口（正常输出或PMOS开漏） 3. LCD分段输出
SEG[42] /PC[3]	O,O	1. 输入口带下拉 2. 输出口（正常输出或PMOS开漏） 3. LCD分段输出

XIN1	I,O	1. 32k晶体输入（32K） 2. 外部RC输入
XOUT1	O,O	32k晶体输出
GND	P	系统接地
VDD1	P	LCD电源供应
VDD	P	LCD电源供应
VDD3	P	LCD电源供应
CUP1	P	电压升压电容器
CUP2	P	电压升压电容器
COM1~4	O	LCD COM输出
COM5/PC[2]	O,I	1. LCD COM输出 2. 输入口带下拉 3. 输出口（正常输出或PMOS开漏）
COM6/PC[1]	O,I	1. LCD COM输出 2. 输入口带下拉 3. 输出口（正常输出或PMOS开漏）
COM7/PC[0]	O,I	1. LCD COM输出 2. 输入口带下拉 3. 输出口（正常输出或PMOS开漏）
COM8/PC[5]+ CAPT2B	O,I	1. LCD COM输出 2. 输入口带下拉 3. 输出口（正常输出或PMOS开漏） 4. 捕捉输入
SEG1~16 /KS1~16	O, O	1. LCD分段输出 2. 按键选通输出
SEG17~22	O	LCD分段输出
SEG23 /PC[6]	O,I/O	1. LCD分段输出 2. PC[6] 输入口带下拉 3. PC[6] 输出口（正常输出或PMOS开漏）
SEG24 /PC[7]	O,I/O	1. LCD分段输出 2. PC[7] 输入口带下拉 3. PC[7] 输出口（正常输出或PMOS开漏）
SEG25 /PE[0]	O,I/O	1. LCD分段输出 2. PE[0] 输入口带下拉 3. PE[0] 输出口（正常输出或PMOS开漏）
SEG26 /PE[1]	O,I/O	1. LCD分段输出 2. PE[1] 输入口带下拉 3. PE[1] 输出口（正常输出或PMOS开漏）

SEG27 /PE[2]	O,I/O	1. LCD分段输出 2. PE[2] 输入口带下拉 3. PE[2] 输出口（正常输出或PMOS开漏）
SEG28 /PE[3]	O,I/O	1. LCD分段输出 2. PE[3] 输入口带下拉 3. PE[3] 输出口（正常输出或PMOS开漏）
SEG29 /PE[4]	O,I/O	1. LCD分段输出 2. PE[4] 输入口带下拉 3. PE[4] 输出口（正常输出或PMOS开漏）
SEG30 /PE[5]	O,I/O	1. LCD分段输出 2. PE[5] 输入口带下拉 3. PE[5] 输出口（正常输出或PMOS开漏）
SEG31 /PE[6]	O,I/O	1. LCD分段输出 2. PE[6] 输入口带下拉 3. PE[6] 输出口（正常输出或PMOS开漏）
SEG32 /PE[7]	O,I/O	1. LCD分段输出 2. PE[7] 输入口带下拉 3. PE[7] 输出口（正常输出或PMOS开漏）
SEG33 /PD[0]	O,I/O	1. LCD分段输出 2. PD[0] 输入口带下拉 3. PD[0] 输出口（正常输出或PMOS开漏）
SEG34 /PD[1]	O,I/O	1. LCD分段输出 2. PD[1] 输入口带下拉 3. PD[1] 输出口（正常输出或PMOS开漏）
SEG35 /PD[2]	O,I/O	1. LCD分段输出 2. PD[2] 输入口带下拉 3. PD[2] 输出口（正常输出或PMOS开漏）
SEG36 /PD[3] /PWM3	O,O	1. LCD分段输出 2. PD[3] 输入口带下拉 3. PD[3] 输出口（正常输出或PMOS开漏） 4. PWM3（TM3）输出
SEG37 /PD[4]+CAPT2A	O,O	1. LCD分段输出 2. PD[4] 输入口带下拉 3. PD[4] 输出口（正常输出或PMOS开漏） 4. CAPT2A输入
SEG38 /PD[5]+INT	O,O	1. LCD分段输出 2. PD[5] 输入口带下拉 3. PD[5] 输出口（正常输出或PMOS开漏）

		4. PIN中断输入
SEG39 /PD[6]/ELP	O,O	1. LCD分段输出 2. PD[6] 输入口带下拉 3. PD[6] 输出口（正常输出或PMOS开漏） 4. ELP输出
SEG40 /PD[7]/ELC	O,O	1. LCD分段输出 2. PD[7] 输入口带下拉 3. PD[7] 输出口（正常输出或PMOS开漏） 4. ELC输出

2. 系统结构

2.1 系统时钟

MK9A50P 有双时钟操作模式，用户可根据要求对配置寄存器的 bit0~3 进行设置。高速时钟的 OSCIN/OSCOUT，通过配置选项可用于连接外部 4MHz 晶体，外部 R 振荡器或内部 500KHz RC 振荡器。当使用者选择内部 500KHz RC 振荡模式时，这两个脚位可作为 KI 或 I/O 口（PA）使用，更具弹性。低速时钟的 XIN/XOUT，情况相同，它们可用于连接外部 32KHz 晶体，外部 R 振荡器或内部低速 RC 振荡器。当使用者选择内部低速 RC 振荡器，这两个脚位可作为输出口使用。时钟模式可通过配置位来选择。一旦双时钟模式被选择，使用者可通过设置 SYS_CTL（$3Eh）的 bit 7 在高、低速时钟间切换，或通过设置 bit0~1 单独开启/关闭这些时钟源。

时钟振荡示意图如下，他们由高速时钟和低速时钟构成。

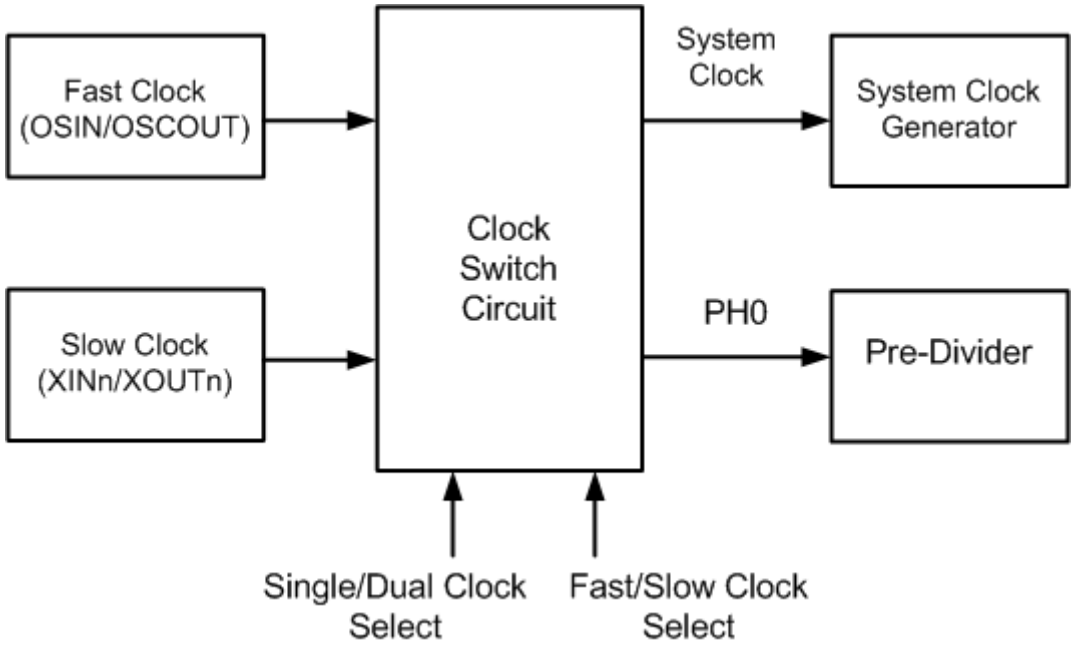


图.2.1.1　时钟交换电路 & 系统时钟产生器

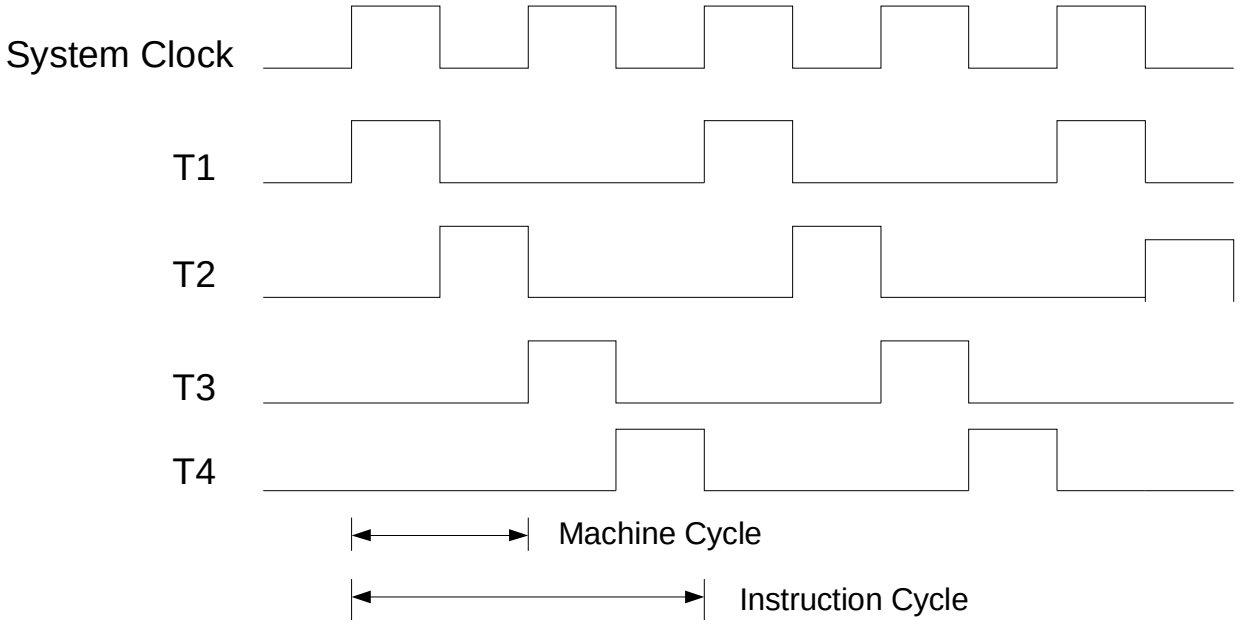


图.2.1.2 机械周期 & 指令周期

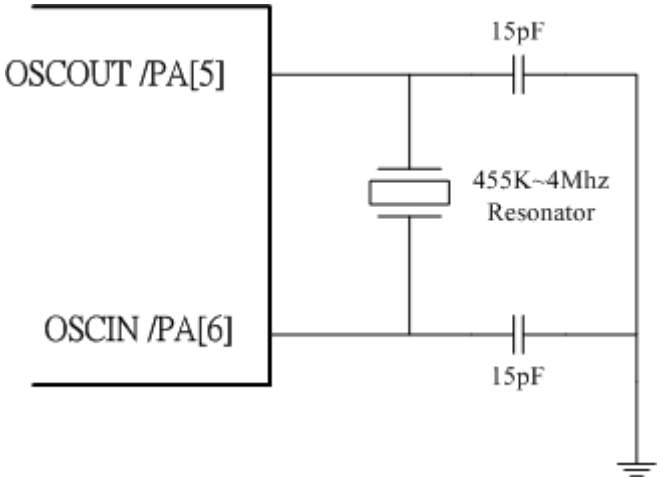
下表显示在不同模式下，系统时钟及预分配器的时钟源状态：

时钟模式	系统时钟	PH0
仅低速时钟	SCLK（低速时钟）	SCLK
仅高速时钟	FCLK（高速时钟）	FCLK
初始阶段（双时钟模式）	SCLK	SCLK
HALT 阶段（双时钟模式）	SCLK	SCLK
低速时钟活动（双时钟模式）	SCLK	SCLK
高速时钟活动（双时钟模式）	FCLK	SCLK

2.1.1 高速时钟（FCLK）连接

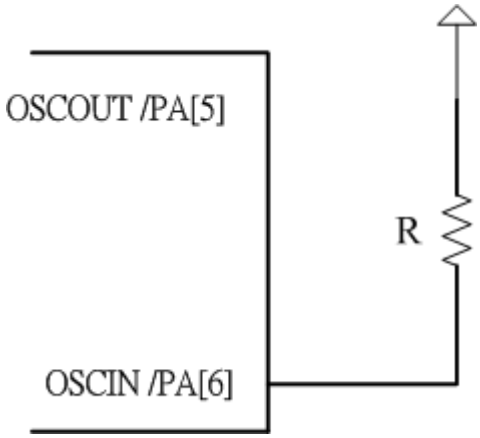
高速时钟有 3 种连接电路，分别是外部最大 4MHz 晶振，外部 R 振荡器及内部高速 RC 振荡器。用户可通过设置配置寄存器 bit 2~3 选择操作模式。连接如下图：

(a) 连接外部 3.58MHz 谐振器，（FOSC1，FOSC0）=（0，0）



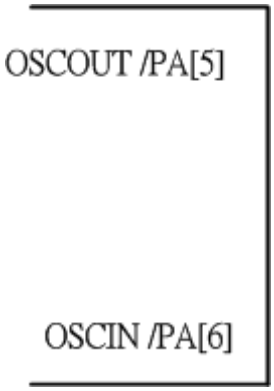
(b) 连接外部 R 振荡器，（FOSC1，FOSC0）=（1，0）

当设置为此模式，OSCOUT 脚位可作为 I/O 口（PA5）使用。



(c) 高速时钟被设置为内部 RC 振荡器或 No，（FOSC1，FOSC0）=（1，1）或（0，1）

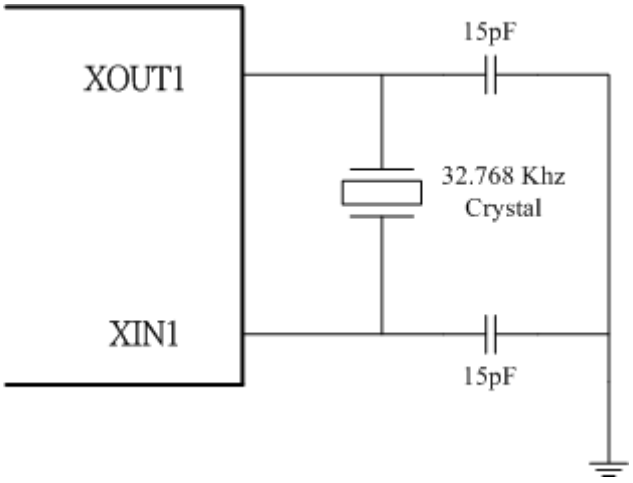
当设置为此模式，OSCIN 及 OSCOUT 可作为 I/O 口（PA6 及 PA5）使用。



2.1.2 低速时钟（SCLK）连接

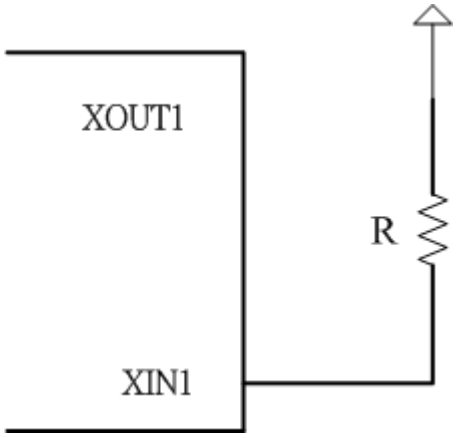
低速时钟有 3 种连接电路，分别是外部 32KHz 晶体，外部 R 振荡器及内部低速 RC 振荡器。使用者可通过设置配置寄存器 bit 0~1 选择操作模式。连接如下图：

(a) 连接外部 32.768KHz 晶体，（SOSC1，SOSC0）=（0，0）



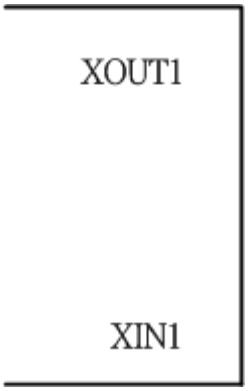
(b) 连接外部 R 振荡器，（SOSC1，SOSC0）=（1，0）

当设置为此模式，XOUT1 脚位**浮动**。



(c) 低速时钟被设置为内部 RC 振荡器或 No，（SOSC1，SOSC0）=（1，1）或（0，1）

当设置为此模式，XIN1 及 XOUT1 **浮动**。



2.1.3 FCLK & SCLK 切换

2.1.3-1 CPU 时钟从 SCLK 切换到 FCLK

```
                                ;; cpu时钟 = SCLK
BC          SYS_CTL,1           ;; 使能高速时钟
NOP                             ;; 高速时钟稳定时间
NOP                             ;; 必需!
NOP                             ;; 必需!
BS          SYS_CTL,7           ;; cpu时钟 = FCLK
```

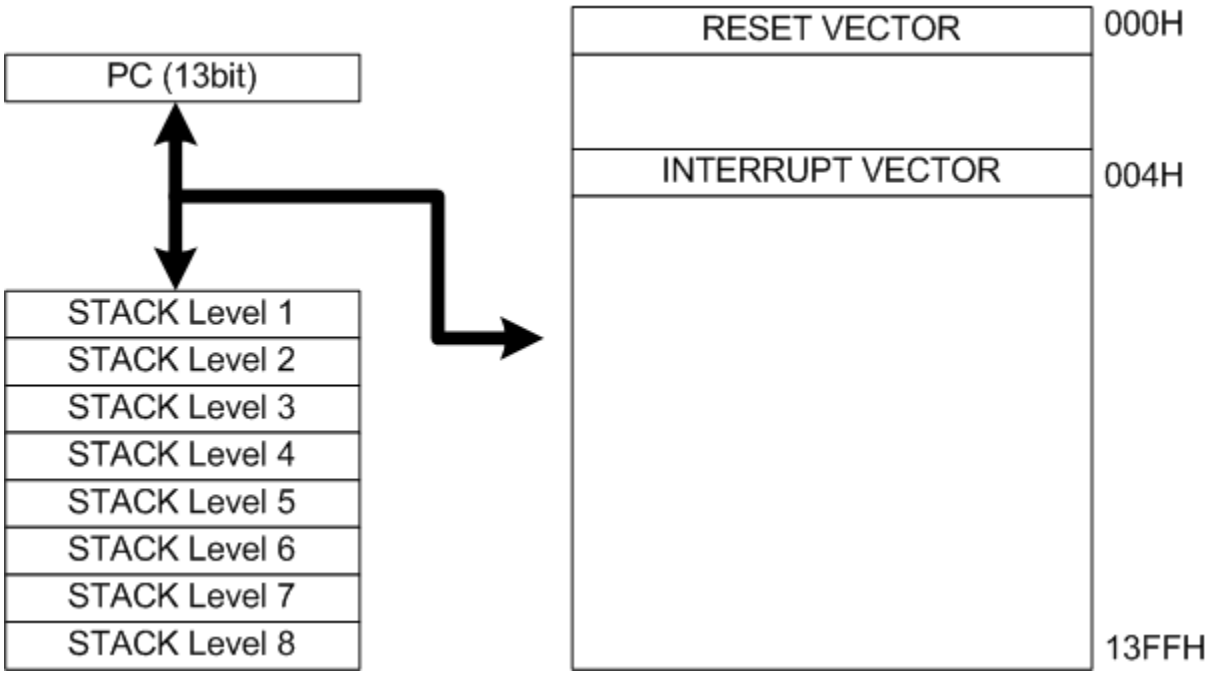
2.1.3-2 CPU 时钟从 FCLK 切换到 SCLK

```
                                ;; FCLK & SCLK全部为ON，cpu时钟 = FCLK
BC          SYS_CTL,7           ;; cpu时钟 = SCLK
NOP                             ;; 必需!
NOP                             ;; 必需!
BS          SYS_CTL,1           ;; 停止FCLK
```

2.2 程序存储器（ROM）

指令及表格被存储在该区域。该存储器只有一个中断向量存在，意味着所有中断发生后都会跳到相同的向量。用户需要使用中断标记去判断是哪一种中断发生。程序计数器（PC）是 13 位，可直接定位所有 13K x 16 位置。查询表格可放置于 ROM 的任何地方。

RESET 向量位于 000H，中断向量位于 004H。如下图：



2.3 数据存储器（RAM）

RAM 容量共由 293 x 8 位组成，包括三种寄存器组。一种是 192 × 8 位操作 RAM，另一种是 59 × 8 位特殊功能寄存器以及 42 x 8 位显示 RAM。数据存储图如下：

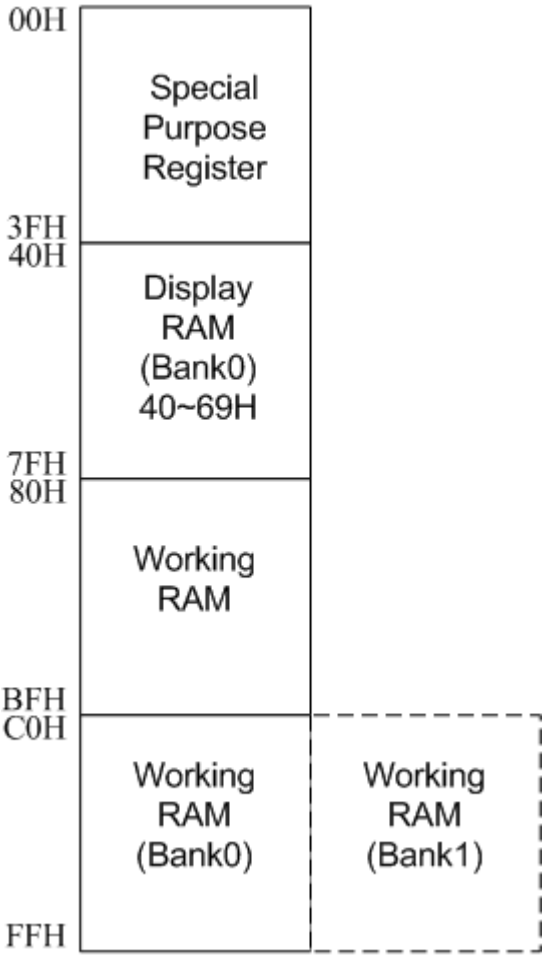


图.2.3.1 存储图

用户可设置 WBANK 寄存器去转换不同数据 RAM 组。

2.4 配置寄存器

此寄存器存储芯片的设计选项，包括复位脚位定义，定时器时钟源选择，LVR 检测电压选择及 WDT 控制。通过软件不能改变寄存器的内容，通过烧录器可固定寄存器的内容。就像我们使用掩膜 ROM 类型 MCU 时的掩膜选项。

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CONFIG_L	LV1	LV0	WDTE	CPT	FOSC1	FOSC0	SOSC1	SOSC0
-	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**
CONFIG_H	--	--	--	WDT_FB	WDT_S	RSTE2	RSTE1	RST_DEF

- Bit 12：看门狗定时器源选择
  - 1：慢看门狗定时器（缺省值是20mS）
  - 0：快看门狗定时器（4个快时钟）
- Bit 11：看门狗定时器源选择
  - 1：低速RC（固定）
  - 0：低速晶体时钟

Bit12	Bit3~2	Bit1~0	Bit11	OSC 类型	看门狗定时器源
WDT_FB	**FOSC1~0**	**SOSC1~0**	**WDT_S**		
1	X X	0 0	0	LP1（低速）	LP晶体
1	X X	0 0	1	LP1（低速）	内部低速RC
1	X X	0 1	X	No	内部低速RC
0	X X	1 0	X	外部 RC	外部低速RC
1	X X	1 1	X	内部 RC	内部低速RC
0	0 0	X X	X	NT	LP晶体
0	0 1	X X	X	No	未使用
0	1 0	X X	X	外部 R	外部高速RC
0	1 1	X X	X	内部 RC	内部 500K RC

- Bit 10~9（RSTE2.1）：PA复位按键号码控制（仅PA作为I/O模式工作）

  当按下PA按键超过2秒，复位发生。

  1 1：最多 4 键复位（同时按下 4 个按键）-- PA7，PA6，PA5，PA3

  1 0：最多 4 键复位（同时按下 4 个按键）-- PA7，PA5，PA4，PA3

  0 1：最多 4 键复位（同时按下 4 个按键）-- PA6，PA5，PA4，PA3

  0 0：最多无按键复位

复位KI/IO	WDTE.RSTE2~1	2'S复位RESET脚位功能	RESET脚位				
			PA3	PA4	PA5	PA6	PA7
KI	X X X	无按键复位功能	X	X	X	X	X
IO	1 X X	无按键复位功能	X	X	X	X	X
IO	0 0 0	无按键复位功能	X	X	X	X	X
IO	0 0 1	PA7，PA6，PA5，PA3	ON	X	ON	ON	ON
IO	0 1 0	PA7，PA5，PA4，PA3	ON	ON	ON	X	ON
IO	0 1 1	PA6，PA5，PA4，PA3	ON	ON	ON	ON	X

<注> 使用此按键复位功能，PA在PAD_CTL2($14)寄存器下首先被设置为I/O口及在正常操作模式下设置PA为输入上拉。当使用此功能进入SLEEP模式，特定PA口将自动设置为上拉。例如，如果用户规定键入PA[0]~PA[1]复位，当系统进入SLEEP模式，PA[0]及PA[1]将被设置为上拉。此时，用户要注意不要按下这些键，否则将产生功耗。

- Bit8（RST_DEF）：RESETB脚位功能定义
  0：RESETB作为正常输入脚位使用
  1：RESETB作为系统复位脚位使用
- Bit7~6（LV1~0）：低电压复位功能电压选择位

Bit7	**Bit6**	**检测电压**
LV1	**LV0**	
0	0	1.5V
0	1	1.7V
1	0	2V
1	1	未使用

- Bit5（WDTE）：看门狗定时器使能/禁止控制
  0：WDT 禁止
  1：WDT 使能
- Bit4（CPT）：密码保护位
  0：ON
  1：OFF
- Bit3~2（FOSC2~1）：OSCIN/OSCOUT频率分配位

Bit3	Bit2	OSC 类型	共振频率
FOSC1	FOSC0		
0	0	NT（正常速度）	455KHz~10Mhz 共振或晶振
0	1	No	OSCIN & OSCOUT 作为 I/O 口或 KI 使用
1	0	外部 R	1. OSCIN 连接到 R（455KHz~4MHz） 2. OSCOUT 可作为 I/O 口使用
1	1	内部 RC	1. 内部高速 RC 振荡器 2. OSCIN & OSCOUT 可作为 I/O 口使用

- Bit1~0（SOSC1~0）：XIN1/XOUT1频率分配位

Bit1	Bit0	OSC 类型	共振频率
SOSC1	SOSC0		
0	0	LP（低速）	32KHz or 64KHz 晶振
0	1	No	XIN1 & XOUT1 浮动
1	0	外部 R	1. XIN1 连接到 R（大约 32KHz） 2. XOUT1 浮动
1	1	内部 RC	1. 内部低速 RC 振荡器 2. XIN1 & XOUT1 浮动

<注> 以下表格列出复位阶段的系统时钟状态及 CLKS 位设置：

Bit3	Bit2	Bit1	Bit0	系统时钟			PH0
FOSC1	FOSC0	SOSC1	SOSC0	Reset	CLKS=0	CLKS=1	
0	0	0	0	低速时钟	低速时钟	高速时钟	低速时钟
0	0	0	1	仅低速时钟，不能写入“1”到 CLKS			低速时钟
0	0	1	X	低速时钟	低速时钟	高速时钟	低速时钟
0	1	0	0	仅高速时钟，不能写入“0”到 CLKS			低速时钟
0	1	0	1	不考虑			XX
0	1	1	X	仅高速时钟，不能写入“0”到 CLKS			高速时钟
1	0	0	0	低速时钟	低速时钟	高速时钟	低速时钟
1	0	0	1	仅低速时钟，不能写入“1”到 CLKS			低速时钟
1	0	1	X	低速时钟	低速时钟	高速时钟	低速时钟
1	1	0	0	低速时钟	低速时钟	高速时钟	低速时钟
1	1	0	1	仅低速时钟，不能写入“1”到 CLKS			低速时钟
1	1	1	X	低速时钟	低速时钟	高速时钟	低速时钟

2.5 特殊功能寄存器

列表寄存器如下，我们将在特定章节中详细描述。

Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INDF	$00	A7	A6	A5	A4	A3	A2	A1	A0
PCL	$01	A7	A6	A5	A4	A3	A2	A1	A0
PCH	$02	--	--	--	A12	A11	A10	A9	A8
STATUS	$03	--	--	--	$\overline{TO}$	$\overline{PD}$	Z	DC	C
FSR	$04	BANK1	BANK0	D5	D4	D3	D2	D1	D0
I/O PAD & Control									
Name	**Addr**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
PA_DIR	$05	--	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PA_CTL	$06	--	KI6/IN6	KI5/IN5	KI4/IN4	KI3/IN3	KI2/IN2	KI1/IN1	KI0/IN0
PA_WAKE_UP	$07	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
PA_EDGE	$2D	EDGE7	EDGE6	EDGE5	EDGE4	EDGE3	EDGE2	EDGE1	EDGE0
PA_PUD1	$08	A3-2	A3-1	A2-2	A2-1	A1-2	A1-1	A0-2	A0-1
PA_PUD2	$09	--	A7-1	A6-2	A6-1	A5-2	A5-1	A4-2	A4-1
PA_DAT	$0A	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PC_WAKE_UP	$1D	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
PC_EDGE	$1E	EDGE7	EDGE6	EDGE5	EDGE4	EDGE3	EDGE2	EDGE1	EDGE0
PC_CTL	$0B	KI7/IN7	KI6/IN6	KI5/IN5	KI4/IN4	KI3/IN3	KI2/IN2	KI1/IN1	KI0/IN0
PC_DIR	$0C	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0
PC_PUD	$0D	UC7	UC6	UC5	UC4	UC3	UC2	UC1	UC0
PC_DAT	$0E	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PD_DIR	$0F	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0
PD_PUD	$10	UD7	UD6	UD5	UD4	UD3	UD2	UD1	UD0
PD_CTL	$11	KI7/IN7	KI6/IN6	KI5/IN5	KI4/IN4	KI3/IN3	KI2/IN2	KI1/IN1	KI0/IN0
PD_DAT	$12	PD7	PD6	PD5	PD4	PD3	PE2	PE1	PD0
PE_DIR	$1A	DE7	DE6	DE5	DE4	DE3	DE2	DE1	DE0
PE_PUD	$1B	UE7	UE6	UE5	UE4	UE3	UE2	UE1	UE0
PE_DAT	$1C	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PAD_CTL1	$13	SEG40/ PD[7]	SEG39/ PD[6]	SEG38/ PD[5]	SEG37/ PD[4]	SEG36/ PD[3]	SEG35/ PD[2]	SEG34/ PD[1]	SEG33/ PD[0]
PAD_CTL2	$14	--	C6	C5	C4	C3	C2	C1	C0
PAD_CTL3	$15	EDGE	--	--	--	--	SEN1_ON	SEN0_ON	REF_ON
PAD_CTL4	$16	SEG32/ PE[7]	SEG31/ PE[6]	SEG30/ PE[5]	SEG29/ PE[4]	SEG28/ PE[3]	SEG27/ PE2]	SEG26/ PE[1]	SEG25/ PE[0]

PAD_CTL5	$28	--	--		SEG23/ PC[6]	SEG24/ PC[7]	SEG42/ PC[3]	SEG41/ PC[4]	PWM3/ PD[3]

TM0: 8-bit Timer (TONE & FREQ out)

Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0_CTL	$17	EN	WR_CNT	DATA	IRQ_S	SUR1	SUR0	DUTY1	DUTY0
TM0_LA	$18	D7	D6	D5	D4	D3	D2	D1	D0
TM0_CNT	$19	D7	D6	D5	D4	D3	D2	D1	D0
TONE_CTL1	$39	EN	PH15E	PH14E	PH13E	PH12E	PH11E	PAT1	INV12
TONE_CTL2	$3A						CRY2	CRY1	CRY0

TM2 : 8-bit Timer x1 , 8-bit capture x1

Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_CTL1	$1F	EN	WR_CNT	BIT	MOD1	MOD0	EDGE	SUR1	SUR0
TM2_CTL2	$20	ENC	CLR_CNT	--	CAPIN1/ RFC_T1	CAPIN0/ RFC_T0	INT_S	PWM_OS	OV
TM2_LA	$21	D7	D6	D5	D4	D3	D2	D1	D0
TM2_CNT	$22	D7	D6	D5	D4	D3	D2	D1	D0
TM3_CTL1	$23	EN	WR_CNT		MOD1	MOD0	EDGE	SUR1	SUR0
TM3_CTL2	$24	ENC	CLR_CNT	--	CAPIN1/ RFC_T1	CAPIN0/ RFC_T0	INT_S	PWM_OS	OV
TM3_LA	$25	D7	D6	D5	D4	D3	D2	D1	D0
TM3_CNT	$26	D7	D6	D5	D4	D3	D2	D1	D0

Interrupt Control

Name	Addr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WBANK	$2E	--	--	--	WKMB0	--	--	--	--
IRQM_CTL	$2F	INTM	--	--	--	--	--	--	--
CPU_RESUME	$30	--	PACR	PINTR	2HZR	PHR	TM3R/ PWM3R/ CAPTR/ RFC3R	TM2R/ PWM2R CAPTR/ RFC2R	TM0R/ TONER
IRQM	$31	--	PACM	PINTM	2HZM	PHM	TM3M/ PWM3M/ CAPT3M/ RFC3M	TM2M/ PWM2M/ CAPT2M/ RFC2M	TM0M/ TONEM
IRQF	$32	--	PACF	PINTF	2HZF	PHF	TM3F/ PWM3F/ CAPT3F RFC3F	TM2F/ PWM2F/ CAPT2F/ RFC2F	TM0F/ TONEF

Other									
LBASDT	$33	LCD1	LCD0	FRAM1	FRAM0	--	DUTY2	DUTY1	DUTY0
STROBE	$34	KIEN1	KIEN0	KOAEN	KOEN	KO3	KO2	KO1	KO0
LCD_CTL	$35	PUMP1	PUMP0	POW1	POW0	OVP1	OVP0	LCDM1	LCDM0
PH_CTL	$36	ELON	EL_SEL	EL_P	CLR	PH_I1	PH_I0	PH_S1	PH_S0
PH_OUT	$37	PH15	PH14	PH13	PH12	PH11	PH10	PH9	PH8
PH_OUT1	$38	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
WDT_CTL	$3B	WDTEN	--	--	--	--	PRE2	PRE1	PRE0
TAB_BNK	$3D	--	--	--	TBA4	TBA3	TBA2	TBA1	TBA0
SYS_CTL	$3E	CLKS	HALT	IRC	LVD1	LVD0	LV	STP1	STP0
ACC	$3F	D7	D6	D5	D4	D3	D2	D1	D0

2.6 HALT 功能

HALT 功能在待机时用于最小能量消耗。在下表用户可设置寄存器$3Eh 的 bit 6 进入 HALT 模式。在此阶段，CPU 操作关闭，意味着程序存储器不工作。只有低速时钟，定时器及 LCD 驱动时钟在工作。

SYS_CTL ($3Eh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SYS_CTL	CLKS	HALT	IRC	LVD1	LVD0	LV	STP1	STP0

Bit	符号	描 述	
7	CLKS	时钟选择 0：低速时钟 1：高速时钟	
6	HALT	CPU打开/关闭控制 0：ON 1：CPU OFF	
5	IRC	内部高速时钟 0：700 Khz（缺省值） 1：1.5 Mhz	
4~3	LVD1~0	低电压检测器	
		1 1	ON（2.56）
		1 0	ON（2.40）
		0 1	ON（2.68）
		0 0	功能OFF
2	LV	低功率输出 0：功率电压 > 2.5V（或2.45V） 1：功率电压 < 2.5V（或2.45V）	
1	STP1	高速时钟控制 0：ON 1：OFF	
0	STP0	低速时钟控制 0：ON 1：OFF	

以下几种情况可释放 HALT 模式：

(1) 脚位改变唤醒（外部中断脚位 PA7~0，PC7~0，PD5）

(2) 捕捉模式（CAPT1A，CAPT1B，CAPT2A & CAPT2B）

(3) 定时器中断（PH，2Hz，TMR0 & TMR2 & TMR3）

(4) 看门狗定时器

(5) 复位

图表如下：

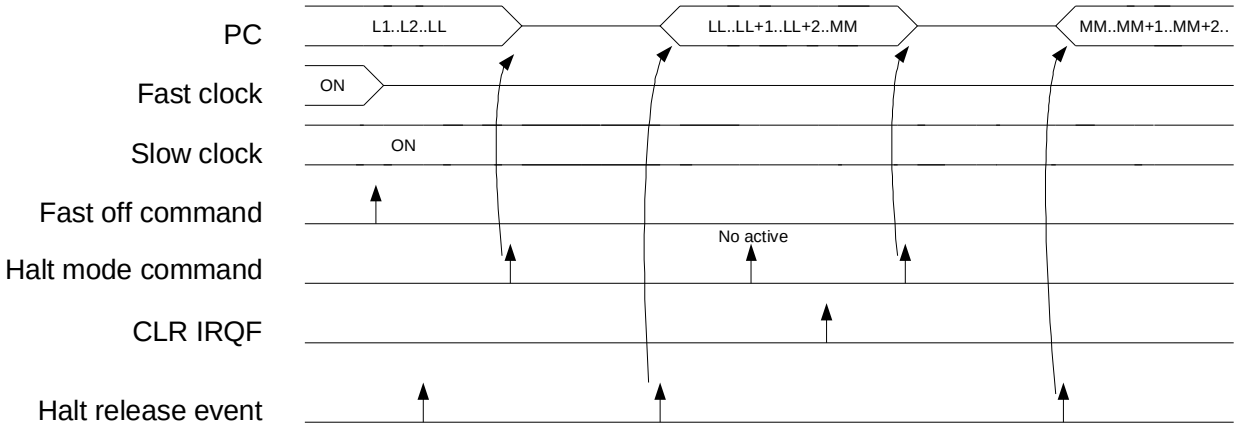


图.2.6.1 HALT 模式 & HALT 释放波形

2.6.1 HALT 模式示例

```
#INCLUDE   "MK9A50P.INC"          ; HALT模式，2hz唤醒
ORG        0x00
           LGOTO     INITIAL

           ORG       004
           MOVLA     B’01101111’
           MOVAM     IRQF          ; 清除PH 2Hz中断标记
           INC       PD_DAT,m
           INC       PA_DAT,m
           IRETI

           ORG       0x100
INITIAL
           CLR       PA_DAT        ; 清除悬空
           CLR       PD_DAT        ; 清除悬空
           MOVLA     0x00          ; 设置PA为输出脚位
           MOVAM     PA_DIR
           MOVLA     0xFF
           MOVAM     PA_PUD1       ; 设置PA为正常输出脚位
           MOVAM     PA_PUD2
           MOVAM     PD_PUD1       ; 设置PD为正常输出脚位
           MOVAM     PD_PUD2
           MOVLA     B'01000111'   ; 帧=42hz，com1~10
           MOVAM     LBASDT
           MOVLA     B'00110010'   ; b5.4=11，低功耗，LCD ON
           MOVAM     LCD_CTL
```

```
                CLR     IRQF        ; 清除中断标记
                MOVLA   B'00010000' ; 设置2HZM中断
                MOVAM   IRQM
                BS      IRQM_CTL,7  ; 使能中断
                BC      SYS_CTL,1   ;; FCLK ON
                NOP
                NOP
                BS      SYS_CTL,7   ;; CPU CLK=FCLK
                ……..
                BC      SYS_CTL,7   ;; CPU CLK= SCLK
LOOP            ……..
                BC      LCD_CTL,1   ; 关闭LCD
                                    ;; 减少功率消耗
                BS      SYS_CTL,6   ; 设置HALT；OSC活动但CPU关闭
                                    ; 如果唤醒，仅系统时钟将被开启
                                    ;; 无加载 & 仅32k，Idd=2.5uA（暂停模式）
                ……..
                LGOTO   LOOP
END
```

2.7 睡眠功能

当通过使用 SLEEP 指令进入睡眠模式，除了看门狗定时器及脚位改变唤醒电路，所有的时钟及电路（包括 LCD）将停止工作。在此模式期间，电流消耗几乎为零。仅两种情况可从 SLEEP 模式唤醒，他们是：

(1) 脚位改变唤醒（外部中断脚位 PA7~0，PC7~0，PD5 & PE4）
(2) 捕捉模式（CAPT1A，CAPT1B，CAPT2A & CAPT2B）
(3) 看门狗定时器
(4) 复位

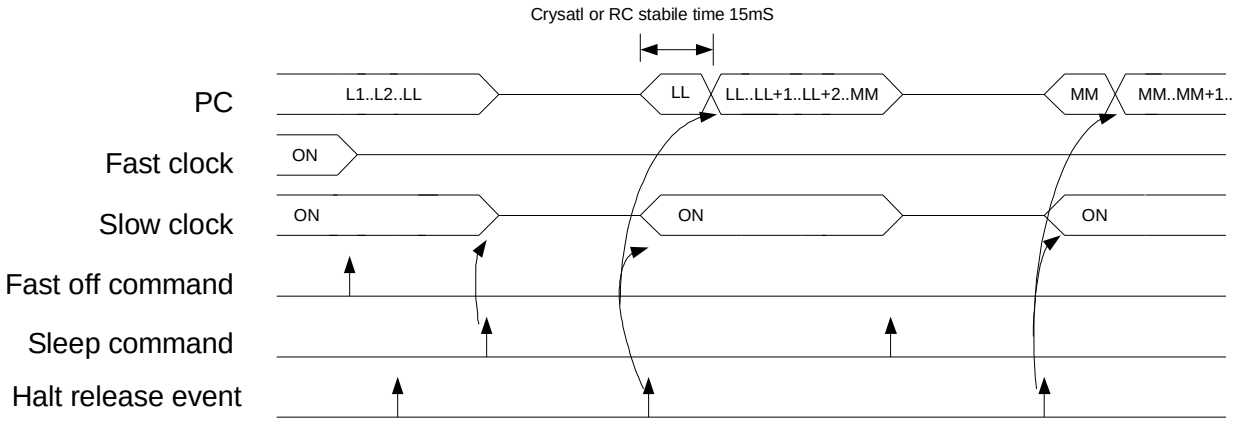


图.2.7.1 SLEEP & HALR 释放波形

2.8 查表功能

MK9A50P提供查表功能。查询表格可置于ROM空间的任何位置。TABRDL指令读取ROM表格的低字节，TABRDH读取高字节。寄存器TAB_BNK及PC7~0可用于定义表格的起始地址。

TAB_BNK ($3Dh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TBH	--	--	--	TBA4	TBA3	TBA2	TBA1	TBA0

- Bit4~0（TBA4~0）：高字节表格定位

<示例>

程序 ROM 地址$700h~$71F OTP 数据转移到工作 RAM（64 bytes）

图.2.8.1 表格读取示例

```
#INCLUDE  "MK9A50P.INC"          ; 转移表格数据到工作RAM（80h~BFh）
#DEFINE   RAM_INDEX C0H
#DEFINE   RAM_DATA  C1H
ORG       0x00
          LGOTO     INITIAL
ORG       0x04
          CLR       IRQF         ; 清除中断标记
          IRETI
          ORG       0x20
INITIAL
          CLR       RAM_INDEX
          CLR       RAM_DATA
          MOVLA     0Ah
          MOVAM     TAB_BNK      ; 计数器
          MOVLA     080h
          MOVAM     FSR          ; 工作RAM 80
TAB:      TABRDH    RAM_INDEX
          NOP
          MOVAM     IAR
          MOV       IAR,a
          INC       FSR,m
          TABRDL    RAM_INDEX
          NOP
          MOVAM     IAR
          MOV       IAR,a
          INC       RAM_INDEX,m
          INC       FSR,m
          BTSS      RAM_INDX,6   ;; A0 ~
```

```
        LGOTO   TAB
        NOP
        sleep
ORG     0A00h
        DW      0001h
        DW      0203h
        DW      0405h
        DW      0607h
        DW      0809h
        DW      0A0Bh
        DW      0C0Dh
        DW      ……
```

2.9 选择寄存器

此寄存器将与INDF寄存器一起用于间接寻址数据存储器。

FSR ($04h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FSR	BANK1	BANK0	D5	D4	D3	D2	D1	D0

Bit	符号	描 述	
7~6	BANK1~0	RAM组选择	
		0 0	特殊功能寄存器
		0 1	显示RAM
		1 0	直接访问工作RAM（80h~BFh）
		1 1	直接访问工作RAM（C0h~DFh）

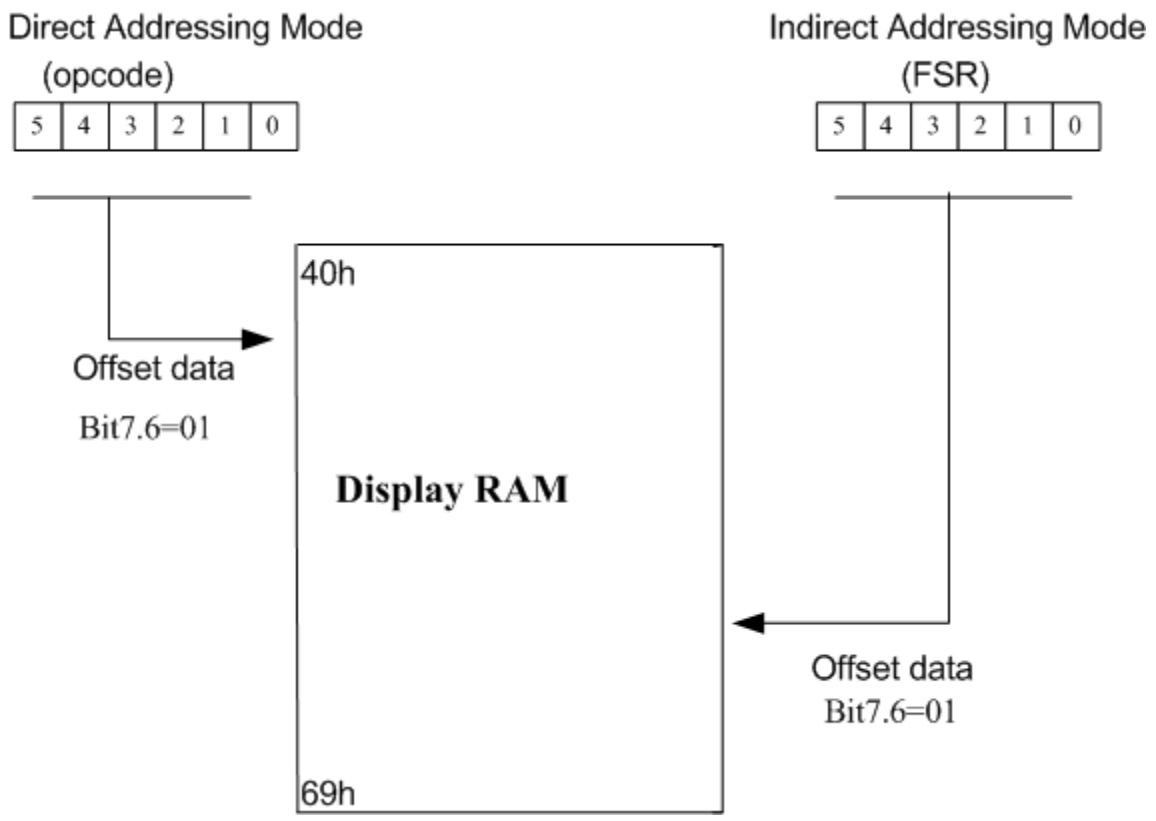


图.2.9.1 工作RAM

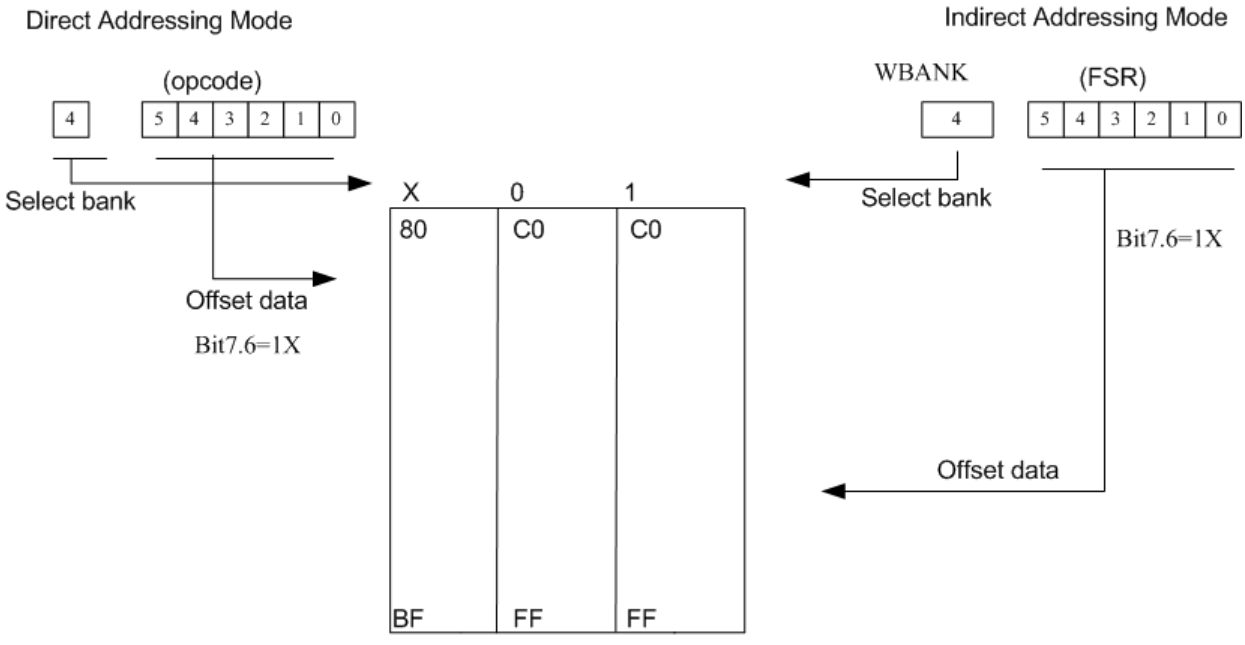


图.2.9.2 显示RAM

2.10 WBANK：RAM 组控制寄存器

该寄存器将与INDF寄存器一起用于间接寻址数据存储

WBANK ($2Eh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WBANK	--	--	--	WKMB0	--	--		--

- Bit4（WKMB0）：工作RAM组选择

Bit4 WKMB0	Bank RAM
0	BANK 0（64 bytes）
1	BANK 1（64 bytes)

2.11 状态寄存器

STATUS寄存器是一个8位寄存器，包含零标记(Z)，进位标记(C)，半进位标记(DC)，电源中断标记($\overline{PD}$)，及看门狗溢出标记($\overline{TO}$)。它记录状态信息。

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 ($\overline{PD}$)：电源中断标记位

$\overline{TO}$	$\overline{PD}$	描 述
0	0	WDT 定时器从睡眠模式溢出（或 4’按键复位）
0	1	WDT 定时器从正常模式溢出（或 4’按键复位）
1	0	睡眠模式下，RESETB 输入一个低电压 睡眠指令
1	1	上电复位 CLRWDT 指令
未改变	未改变	睡眠模式下，RESETB 输入一个低电压

- Bit2 (Z)：零标记位
  - 0：逻辑操作结果不是零
  - 1：逻辑操作结果是零
- Bit1 (DC)：半进位及半借位标记位

ADD 指令：

- 0：无进位
- 1：从低四位进位

SUB指令：

- 0：从低四位借位
- 1：无借位

- Bit0 (C)：进位及借位标记位

ADD 指令：

- 0：无进位
- 1：从 MSB 进位

SUB指令：

- 0：从MSB借位
- 1：无借位

<table>
<tr><th>Bit</th><th>符号</th><th colspan="2">描 述</th></tr>
<tr><td>4</td><td>$\overline{TO}$</td><td colspan="2">超时标记位：
1：上电后或通过CLRWDT或SLEEP指令
0：发生看门狗定时器溢出</td></tr>
<tr><td>3</td><td>$\overline{PD}$</td><td colspan="2">电源中断标记位：(注2)
1：上电后或通过CLRWDT指令
0：执行SLEEP指令</td></tr>
<tr><td>2</td><td>Z</td><td colspan="2">零位：
1：逻辑操作结果是零
0：逻辑操作结果不是零</td></tr>
<tr><td rowspan="3">1</td><td rowspan="3">DC</td><td colspan="2">半进位及半借位</td></tr>
<tr><td>ADD指令</td><td>SUB指令</td></tr>
<tr><td>1：从低四位进位
0：无进位</td><td>1：无借位
0：从低四位借位</td></tr>
<tr><td rowspan="3">0</td><td rowspan="3">C</td><td colspan="2">进位及借位</td></tr>
<tr><td>ADD指令</td><td>SUB指令</td></tr>
<tr><td>1：从 MSB 进位
0：无进位</td><td>1：无借位 (注1)
0：从MSB借位</td></tr>
</table>

2.12 PCH & PCL

PCH ($02h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCH	--	--	--	A12	A11	A10	A9	A8

PCL ($01h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	A7	A6	A5	A4	A3	A2	A1	A0

MK9A50P 有一个 12 位程序计数器（PC），包括 PCL（8 位）及 PCH（4 位）。PC 用于存储路由程序。当用户改变 PCL 值，则程序将会跳到指示位。

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

Ex2：PCH= 01H，PCL=F0H+30H=20H 带进位 1，程序将会跳到 PC=220H，但 PCH 始终为 01H。

<注>

(a) 当执行 IRET 及 IRETI 时，PCH 数据将不会更新。

(b) 当执行 RETLW，LGOTO，LCALL 及 RET 时，PCH 将会更新。

(c) 当数学操作后及 PC[8]被改变后，PCH 将会被更新。

<示例>

图.2.12.1 PCL & PCL 控制示例

以下程序表明 PCL 及 PCH 在直接数学下如何工作。

```
#DEFINE    PCL      01H        ; 定义地址，RAM 的 01H 为 PCL
#DEFINE    PCH      02H        ; 定义地址，RAM 的 02H 为 PCH

           ORG      00
           LGOTO    START
           ORG      1C0h
START:     MOVLA    02h
           MOVAM    PCH
           MOVLA    33H
           MOVAM    PCL
           NOP
           NOP
           NOP
           NOP
           ORG      233h
           LGOTO    A1
```

```
        NOP
A1:     ORG     2FCh
        MOVLA   04h
        MOVAM   PCH
        MOVLA   88H
        MOVAM   PCL
        NOP
        NOP
        MOVLA   80h
        ADD     PCL,m
        NOP
        NOP
        MOVLA   A0h
        SUB     PCL,m
        NOP
        NOP
        MOVLA   020h
        NOP
        ORG     36Ch
        NOP
        MOVLA   022h
        NOP
        ORG     370
        NOP
        MOVLA   024h
        NOP
        NOP
        NOP
        ORG     388h
        NOP
        MOVLA   026h
        NOP
        MOVLA   80h
        ADD     PCL,m
        NOP
        NOP
        ORG     40Dh
        NOP
        MOVLA   028h
```

```
        NOP
        MOVLA   0A0h
        SUB     PCL,m
        NOP
        MOVLA   02Ah
        NOP
End
```

图.2.12.2 程序流程

以下程序表明 PCL 及 PCH 在直接数学下如何工作。

```
Current PC      ORG     1C0h        指令被执行后 PC 定位
1C0             MOVLA   02h         ; PC=1C1H，PCL =C1H，PCH=00H。
1C1             MOVAM   PCH         ; PC=1C2H，PCL =C2H，PCH=02H。
1C2             MOVLA   33H         ; PC=1C3H，PCL =C3H，PCH=02H。
1C3             MOVAM   PCL         ; PC=233H，PCL =33H，PCH=02H。
                                    ; 程序将会跳到 PC=233H
233             LGOTO   A1          ; PC=2FCH，PCL =FCH，PCH=02H。
2FC     A1:     MOVLA   04h         ; PC=2FDH，PCL =FDH，PCH=02H。
2FD             MOVAM   PCH         ; PC=2FEH，PCL =FEH，PCH=04H。
                MOVLA   88H
2FE                                 ; PC=300H，PCL =C0H，PCH=04H。
                                      当 PCH=FFh è 00h，PCH ç PC[11:8]
                                    ;; 错误!!
2FF             MOVAM   PCL         ; PC=388H，PCL =88H，PCH=03H。（PCH ç PC[11:8]+1）
                                    ; 程序将会跳到 PC=388H
388             NOP                 ; PC=389H，PCL =89H，PCH=03H。
389             MOVLA   26h         ; PC=38AH，PCL =8AH，PCH=03H。
38A             NOP                 ; PC=38BH，PCL =8BH，PCH=03H。
38B             MOVLA   80h         ; PC=38CH，PCL =8CH，PCH=03H。
38C             ADD     PCL         ; PC=40DH，PCL =0DH，PCH=04H。
40D             NOP                 ; PC=40EH，PCL =0EH，PCH=04H。
40E             MOVLA   28h         ; PC=40FH，PCL =0FH，PCH=04H。
40F             NOP                 ; PC=410H，PCL =10H，PCH=04H。
410             MOVLA   A0h         ; PC=411H，PCL =11H，PCH=04H。
411             SUB     PCL         ; PC=372H，PCL =72H，PCH=03H。
372             NOP                 ; PC=373H，PCL =73H，PCH=03H。
373             NOP
```

2.13 复位

4种状况将会引起复位，分列如下。电源中断将会引起MK9A50P复位，检测电压取决于CONFIG寄存器的bit7~bit6。此条件用于缺电状态下保护芯片。最后两种情况称为暖复位。不同的复位状况将会影响寄存器及RAM。$\overline{TO}$ 及 $\overline{PD}$ 位可用于决定复位类型。

(1) 上电复位。(冷复位)

(2) 低电压复位（LVR)。(冷复位)

(3) RESETB脚位复位（输入一个负脉冲)。(热复位)

(4) 2'S按键复位。(热复位)

(5) WDT定时器溢出复位。(热复位)

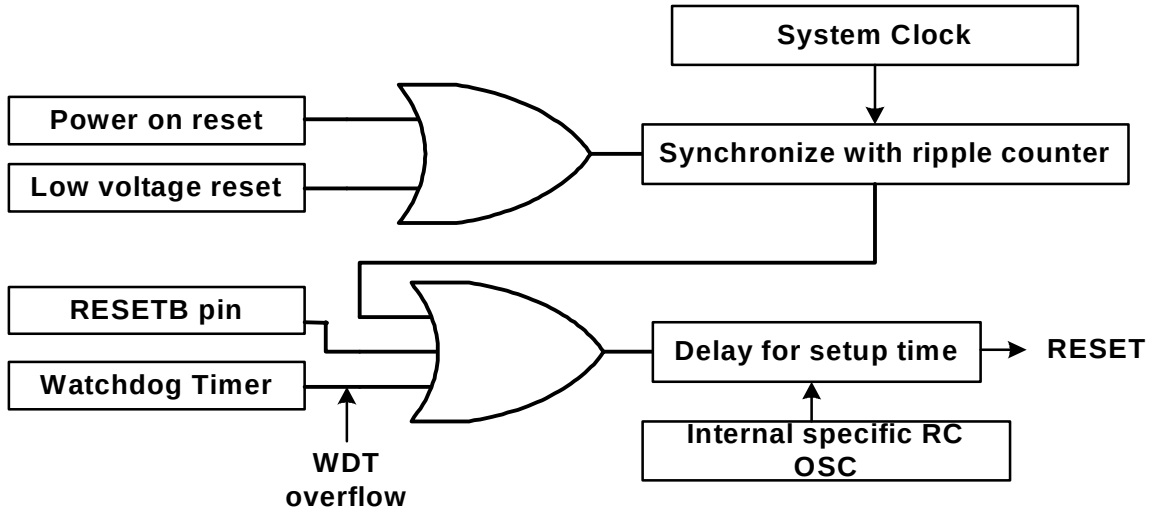


图.2.13.1 复位图

<注>：看门狗设置时间为大约20ms，由于电源电压，过程及温度变化原因会存在些误差。

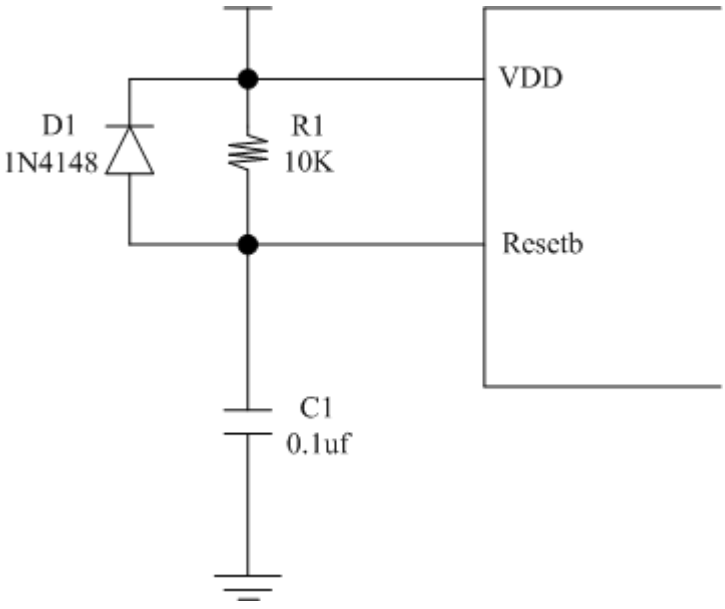


图.2.13.2 复位电路

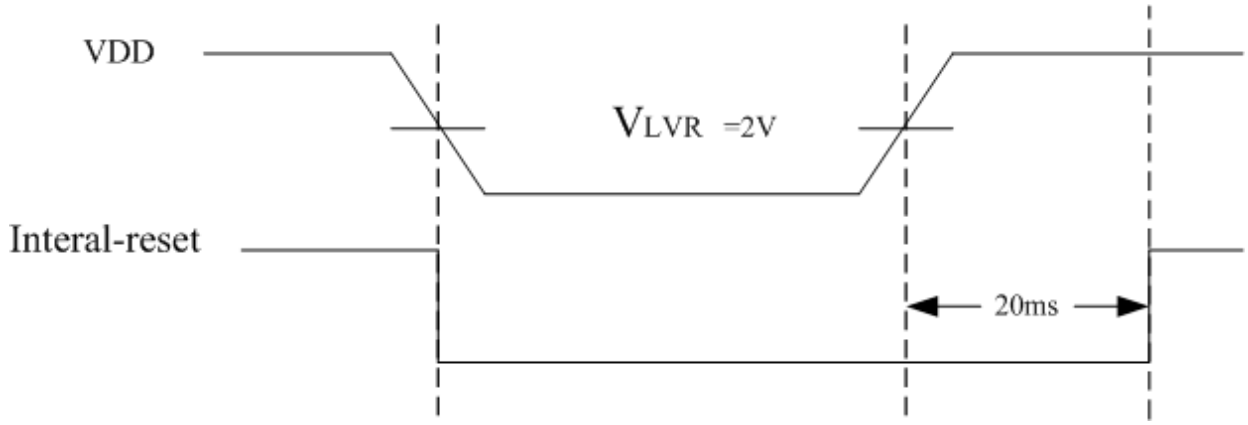


图.2.13.3 LVR ON

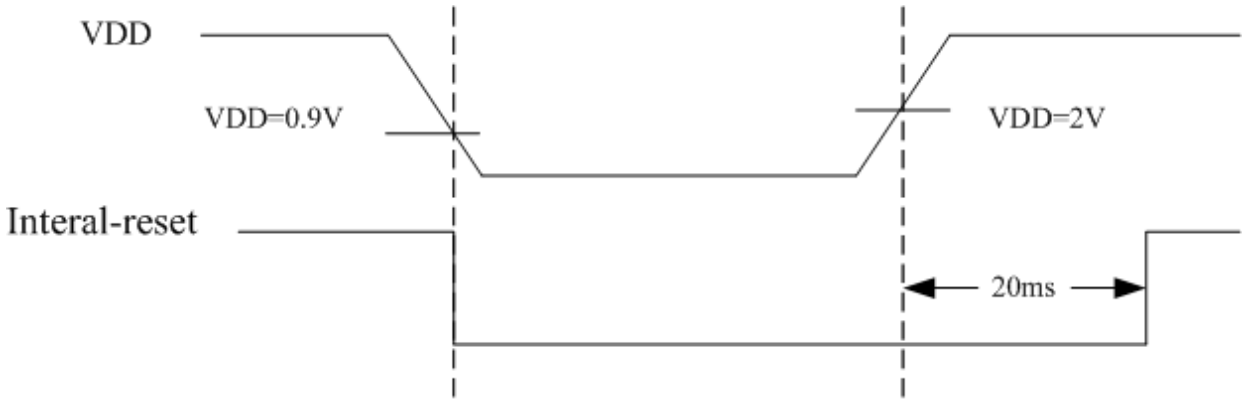


图.2.13.4 LVR OFF

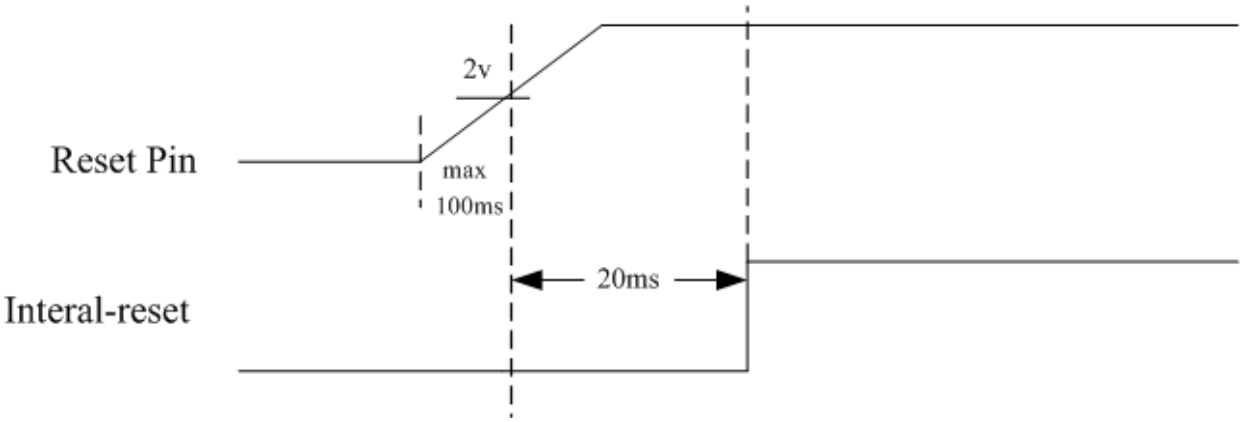


图.2.13.5 外部复位 -- RESETB PIN

不同复位条件下的缺省值

Address	Name	Cold Reset	Warm Reset
00H	INDF	0000 0000	0000 0000
01H	PCL	0000 0000	0000 0000
02H	PCH	- - - 0 0000	- - - 0 0000
03H	STATUS	0001 1XXX	0001 1PPP
04H	FSR	0000 0000	0000 0000
05H	PA_DIR	0111 1111	0111 1111
06H	PA_CTL	0000 0000	0000 0000
07H	PA_WAKE_UP	0000 0000	0000 0000
2DH	PA_EDGE	0000 0000	0000 0000
08H	PA_PUD1	0000 0000	0000 0000
09H	PA_PUD2	0000 0000	0000 0000
0AH	PA_DAT	XXXX XXXX	PPPP PPPP
1DH	PC_WAKE_UP	0000 0000	0000 0000
1EH	PC_EDGE	0000 0000	0000 0000
0BH	PC_CTL	0000 0000	0000 0000
0CH	PC_DIR	1111 1111	1111 1111
0DH	PC_PUD	0000 0000	0000 0000
0EH	PC_DAT	0000 0000	0000 0000
0FH	PD_DIR	1111 1111	1111 1111
10H	PD_PUD	0000 0000	0000 0000
11H	PD_CTL	0000 0000	0000 0000
12H	PD_DAT	0000 0000	0000 0000
1AH	PE_DIR	1111 1111	1111 1111
1BH	PE_PUD	0000 0000	0000 0000
1CH	PE_DAT	0000 0000	0000 0000
13H	PAD_CTL1	0000 0000	0000 0000
14H	PAD_CTL2	0000 0000	0000 0000
15H	PAD_CTL3	0000 0000	0000 0000
16H	PAD_CTL4	0000 0000	0000 0000
28H	PAD_CTL5	0000 0000	0000 0000
17H	TM0_CTL	0000 0000	0000 0000
18H	TM0_LA	0000 0000	0000 0000
19H	TM0_CNT	0000 0000	0000 0000
39H	TONE_CTL1	0000 0000	0000 0000
3AH	TONE_CTL2	0000 0000	0000 0000

1FH	**TM2_CTL1**	**0000 0000**	**0000 0000**
20H	**TM2_CTL2**	**0000 0000**	**0000 0000**
21H	**TM2_LA**	**0000 0000**	**0000 0000**
22H	**TM2_CNT**	**0000 0000**	**0000 0000**
23H	**TM3_CTL1**	**0000 0000**	**0000 0000**
24H	**TM3_CTL2**	**0000 0000**	**0000 0000**
25H	**TM3_LA**	**0000 0000**	**0000 0000**
26H	**TM3_CNT**	**0000 0000**	**0000 0000**
2EH	**WBANK**	**0000 0000**	**0000 0000**
2FH	**IRQM_CTL**	**0000 0000**	**0000 0000**
30H	**CPU_RESUME**	**0000 0000**	**0000 0000**
31H	**IRQM**	**0000 0000**	**0000 0000**
32H	**IRQF**	**0000 0000**	**0000 0000**
33H	**LBASDT**	**0101 0010**	**0101 0010**
34H	**STROBE**	**0000 0000**	**0000 0000**
35H	**LCD_CTL**	**0000 0000**	**0000 0000**
36H	**PH_CTL**	**0000 0000**	**0000 0000**
37H	**PH_OUT**	**XXXX XXXX**	**PPPP PPPP**
38H	**PH_OUT1**	**XXXX XXXX**	**PPPP PPPP**
3BH	**WDT_CTL**	**1000 0111**	**1000 0111**
3DH	**TAB_BNK**	**0000 0000**	**0000 0000**
3EH	**SYS_CTL**	**0000 0010**	**0000 0010**
3FH	**ACC (Accumulator)**	**XXXX XXXX**	**PPPP PPPP**

X：未知；　　**?**：数值取决于条件；

P：不变；　　**-** ：未实施，读为“**0**”

3. 定时器与捕捉

3.1. 16 位预分配器

16 位预分配器可产生 2Hz 定时器捕捉，它可提供一个精确的 0.5 秒时基。另外，其中一些预分配定时器将会变成许多其他时基，也就是，LCD 驱动器，定时器及捕捉。有两个寄存器用于指示 16 位分配器触发器的每一阶段状态，从 PH 可同步化为一个时基。这两个寄存器可被读出。

PH_OUT ($37h) & PH_OUT1($38h): （只读）

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PH_OUT	PH15	PH14	PH13	PH12	PH11	PH10	PH9	PH8
PH_OUT1	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PH_OUT	1hz	2hz	4hz	8hz	16hz	32hz	64hz	128hz
PH_OUT1	256hz	512hz	1K	2K	4K	8K	16K	32K

3.2. 定时器 0（TM0）

芯片的定时器0有两个功能，一个是作为通用8位定时器，另一个是作为FREQ或TONE的时基。TM0有两个缓冲器，TM0_LA 及 TM0_CNT。在定时器 0 开始计数前，用户应该写计数器数值到 TM0_LA。如果 WR_CNT（TM0_CTL 的 Bit 6）被设置为“1”，数据将会自动下载到 TM0_CNT。此设置在第一次时是必需的。当定时器正在计数，溢出发生之后，它将产生中断，自动重载功能将重载 TM0_LA 数据到 TM0_CNT。方块图如下：（图.3.2.1）

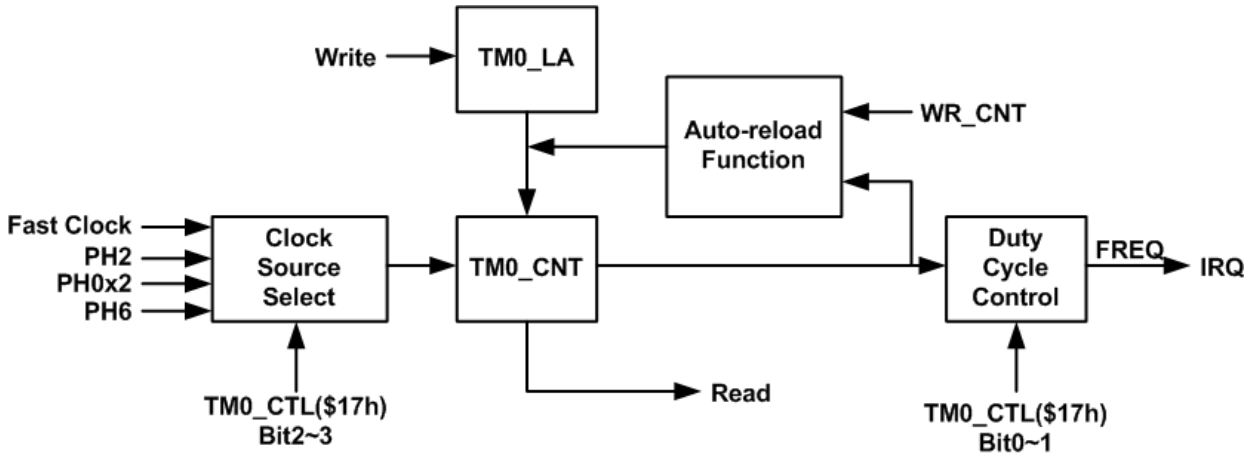


图.3.2.1 定时器 0（TM0）示意图

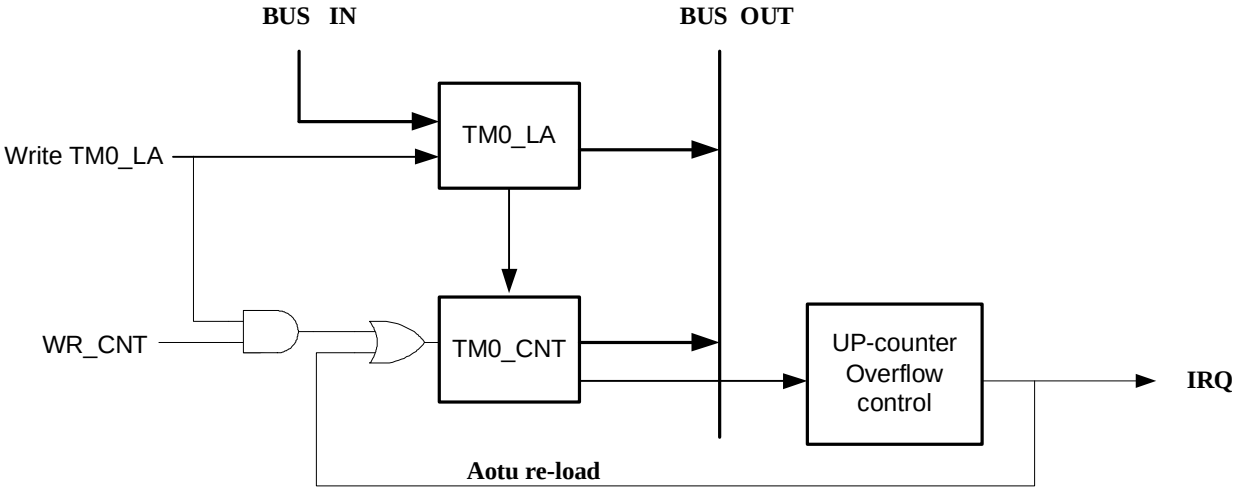


图.3.2.2 定时器 0 WR_CNT & 自动重载控制

如果 TM0 作为通用 8 位定时器使用，用户可通过设置 TM0_CTL($17h)寄存器的 bit 2~3 来选择时钟源。占空比控制时钟的缺省值是（0，0），即效率为 1：1。因为 TM0 将会用于产生 FREQ 或音调，占空比可能改变成另一个数值。一旦用户要转换这些功能，在再次使用 TM0 作为通用定时器时，请注意这两位的数据。

3.3. FREQ 及 TONE 发生器

TM0 也可以用于产生 FREQ 及 TONE。该图由 TM0 示意图延伸而来。请参考下图：

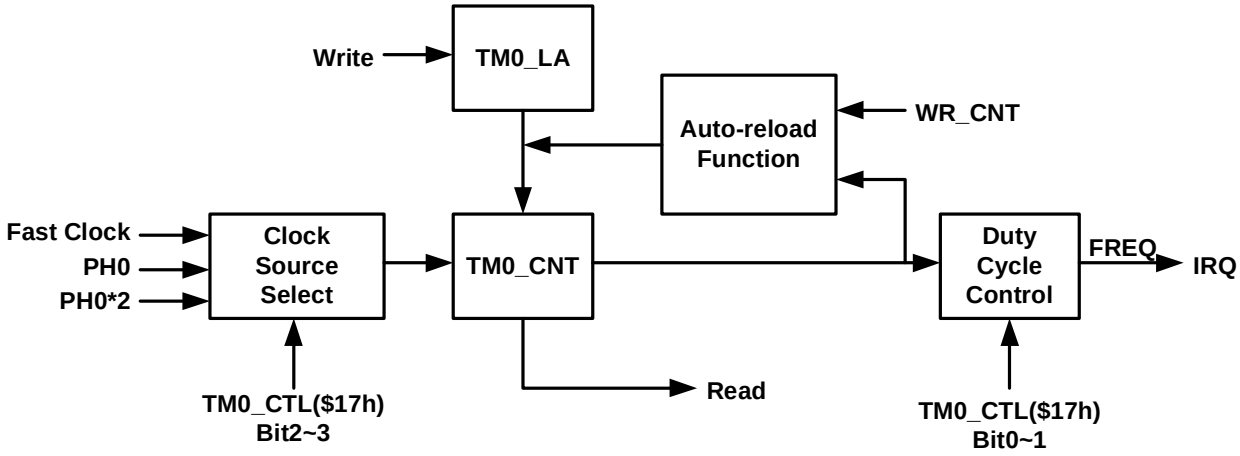


图.3.3.1 FREQ 示意图

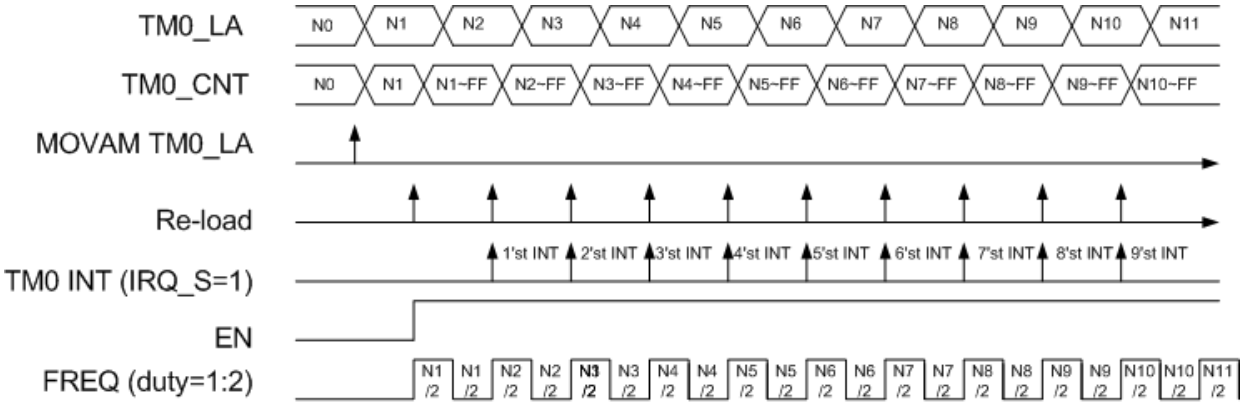


图.3.3.2 FREQ 1:1 & INT 波形

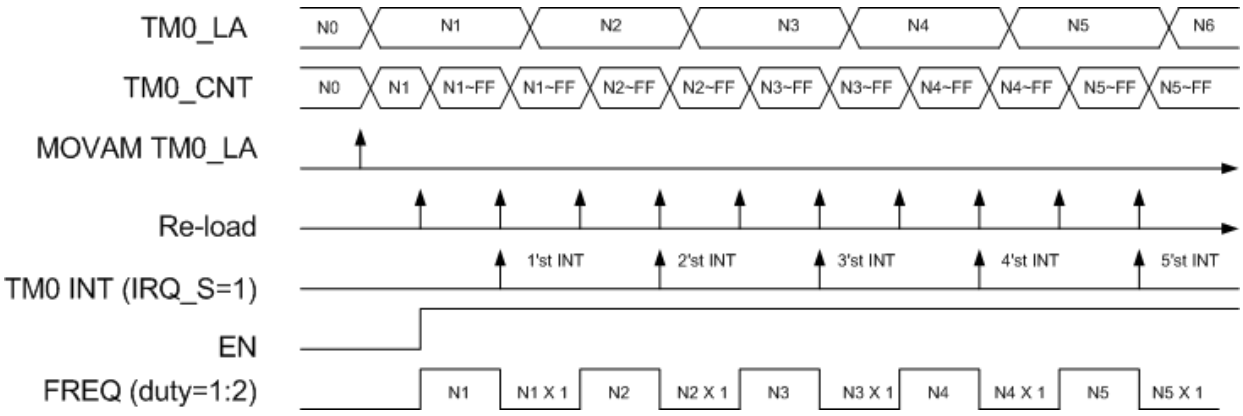


图.3.3.3 FREQ 1:2 & INT 波形

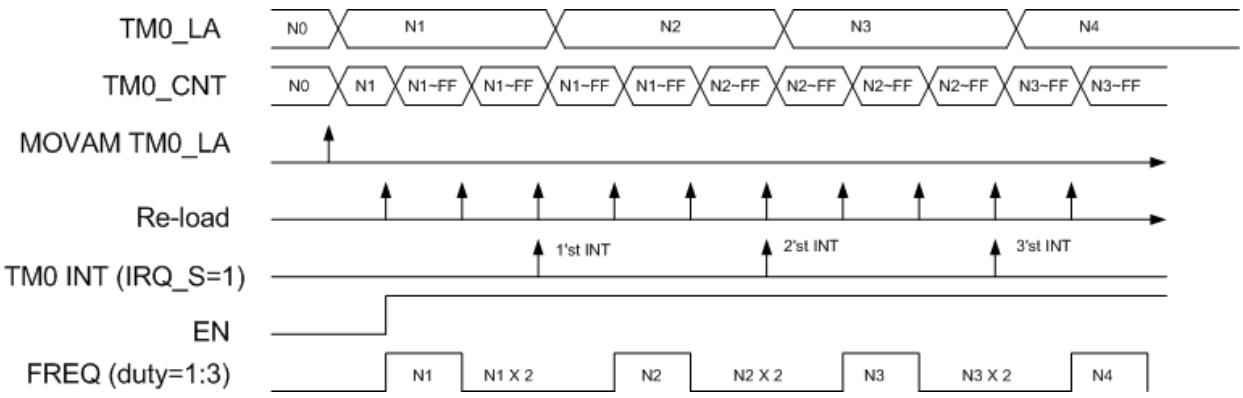


图.3.3.4 FREQ 1:3 & INT 波形

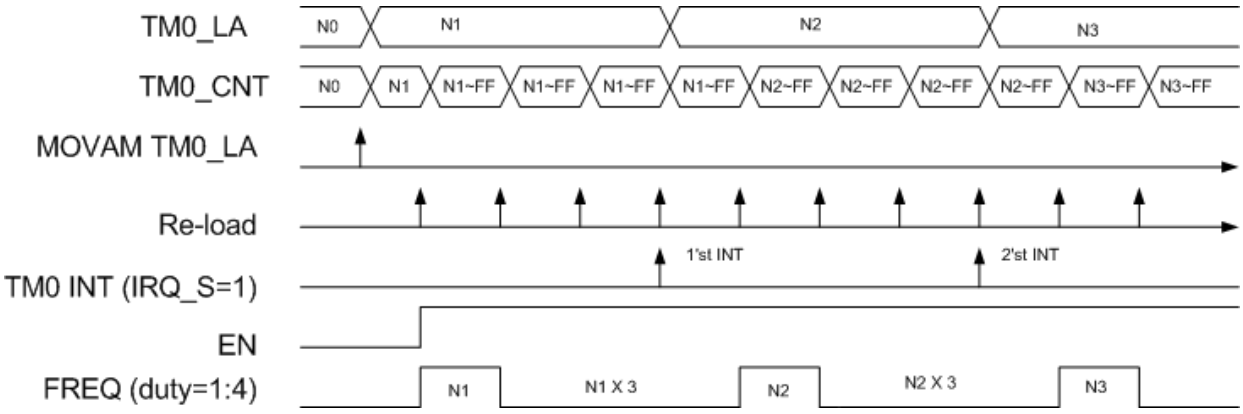


图.3.3.5 FREQ 1:4 & INT 波形

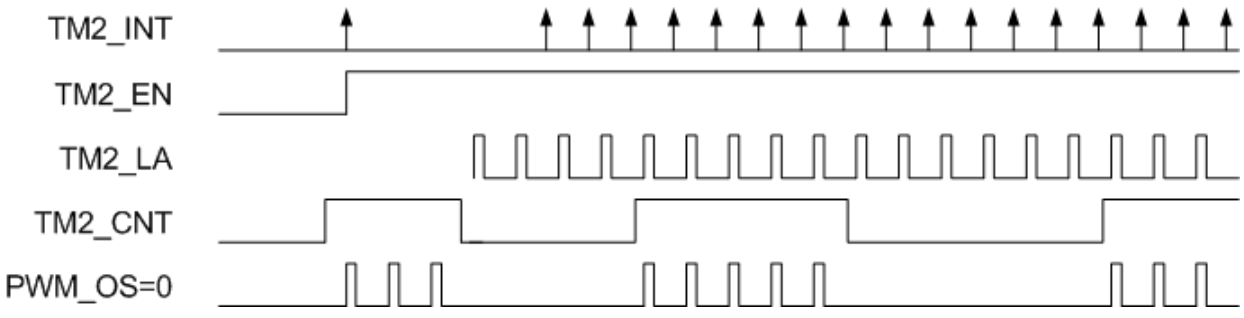


图.3.3.6 远程 & FREQ 波形

FREAOUT 脚位频率与 FREQ 相同。请参考下面相关寄存器定义及设置流程。

通过设置下表中的一些寄存器，BZ 输出脚位源可以是 FREQ 或 TONE。因为 FREQ 也可以是一些时钟的时基，所以当 TM0 活动时它总会存在。

TM0_CTL ($17h)：TM0 控制

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0_CTL	EN	WR_CNT	DATA	IRQ_S	SUR1	SUR0	DUTY1	DUTY0

Bit	符号	描 述	
7	EN	TM0使能/禁止 0：禁止 1：使能	
6	WR_CNT	TM0_CNT将通过写数据到TM0_LA被设置 0：禁止 1：使能	
5	DATA	远程模式，数据输出。（TM0 作为载体工作）	
4	**IRQ_S**	TM0 中断输出 0：TM0 溢出中断（RFC 模式使用） 1：FREQ 循环中断（远程 & FREQ 使用）	
3~2	SUR1~0	SUR1~0：TM0时钟源选择	
		0 0	高速时钟
		0 1	PH2
		1 0	PH0 X 2
		1 1	Ph6
1~0	DUTY1~0	DUTY1~0：占空比	
		0 0	1 : 1　H脉冲 : L脉冲=T/2 : T/2
		0 1	1 : 2　H脉冲 : L脉冲=T : T
		1 0	1 : 3　H脉冲 : L脉冲=T : 2T
		1 1	1 : 4　H脉冲 : L脉冲=T : 3T

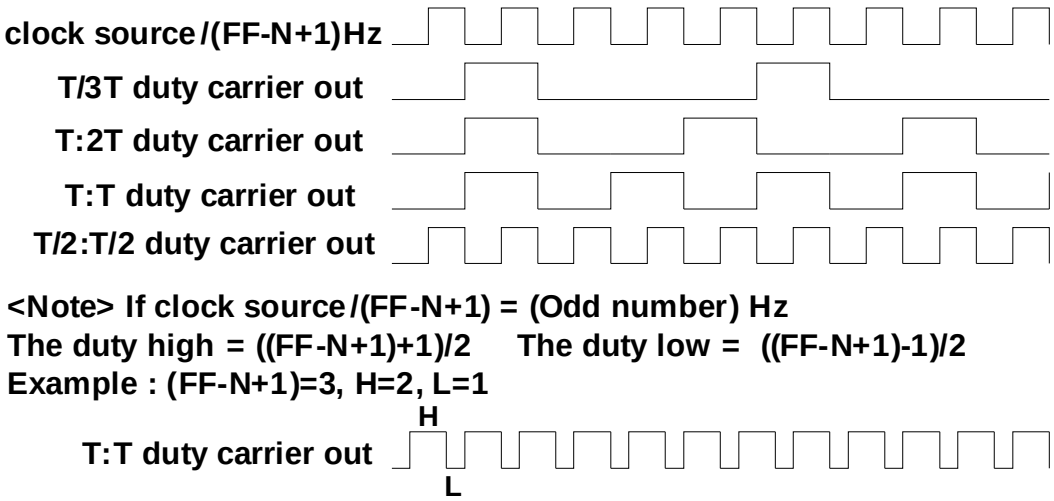


图.3.3.7 占空比设置定时图*（N =/= FFh）*

TM0_LA ($18h)：TM0 数据（R/W）

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0_LA	D7	D6	D5	D4	D3	D2	D1	D0

l Bit7~0：定时器 0 锁存数据*（Data =/= FFh）*

TM0_CNT($19h)：TM0 计数器（R）（上计数器）

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0_CNT	D7	D6	D5	D4	D3	D2	D1	D0

l Bit7~0：定时器 0 计数器数据

<注> 1. 该寄存器是只读寄存器

2. TM0 是上计数定时器

3. 带自动重载功能

图.3.3.5 远程输出示例

```
            INC     'MK9A50P.inc'   ;; 远程控制
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x7E            ;; 清除 TM0 IRQF
            MOVAM   IRQF
            INC     RAM_*0,m        ;; 语音载体控制
            BTSS    RAM_80,3
            LGOTO   DATA_HI
            BC      TM0_CTL,5       ;; 远程低输出
            LGOTO   DATA_END
DATA_HI     BS      TM0_CTL,5       ;; 远程高输出
            CLR     RAM_80
DATA_END    NOP
            IRETI
```

```
        ORG     100h
START:  CLR     STATUS
        BC      SYS_CTL,b1      ;; 快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
        CLR     PAD_CTL1        ;; PD7~0 被选择
        CLR     PD_DIR          ;; PD7~0 输出
        MOVLA   B'0101111'      ;; ra3-远程输出
        PAD_CTL2
        MOVLA   B'01010010'     ;; 快时钟，H:L=T:2T=1/3
        MOVAM   TM0_CTL
        MOVLA   B'11101001'     ;;4Mhz/[(FF-E9+1)x4]=4096K/72=56.9K
        MOVAM   TM0_LA
        MOVLA   0x01            ;; TM0 IRQ 掩膜
        MOVAM   IRQM
        BC      TM0_CTL,6       ;; 禁止透写
        BS      TM0_CTL,5
        BS      IRQM_CTL,7
        BS      TM0_CTL,7
        LGOTO   $
```

TONE_CTL1 ($39h): (W)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TONE_CTL1	EN	PH15E	PH14E	PH13E	PH12E	PH11E	PAT1	INV12

- Bit7：TONE 使能信号
  0：TONE 禁止
  1：TONE 使能
- Bit6：PH15 使能/禁止
  0：禁止
  1：使能
- Bit5：PH14 使能/禁止
  0：禁止
  1：使能
- Bit4：PH13 使能/禁止
  0：禁止
  1：使能

- Bit3：PH12 使能/禁止
  0：禁止
  1：使能
- Bit2：PH11 使能/禁止
  0：禁止
  1：使能
- Bit1：（PH14 及 PH13）使能/禁止
  0：禁止
  1：使能
- Bit0：延迟 1/16 秒使能/禁止
  0：禁止
  1：使能

TONE_CTL2 ($3Ah): (W)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TONE_CTL2						CRY2	CRY1	CRY0

- Bit2-0：载体选择信号
  000：FREQOUT 载体
  001：1Khz 载体
  010：2Khz 载体
  011：4Khz 载体
  100：PWM2（TM2 PWM 输出）
  101：PWM3（TM3 PWM 输出）

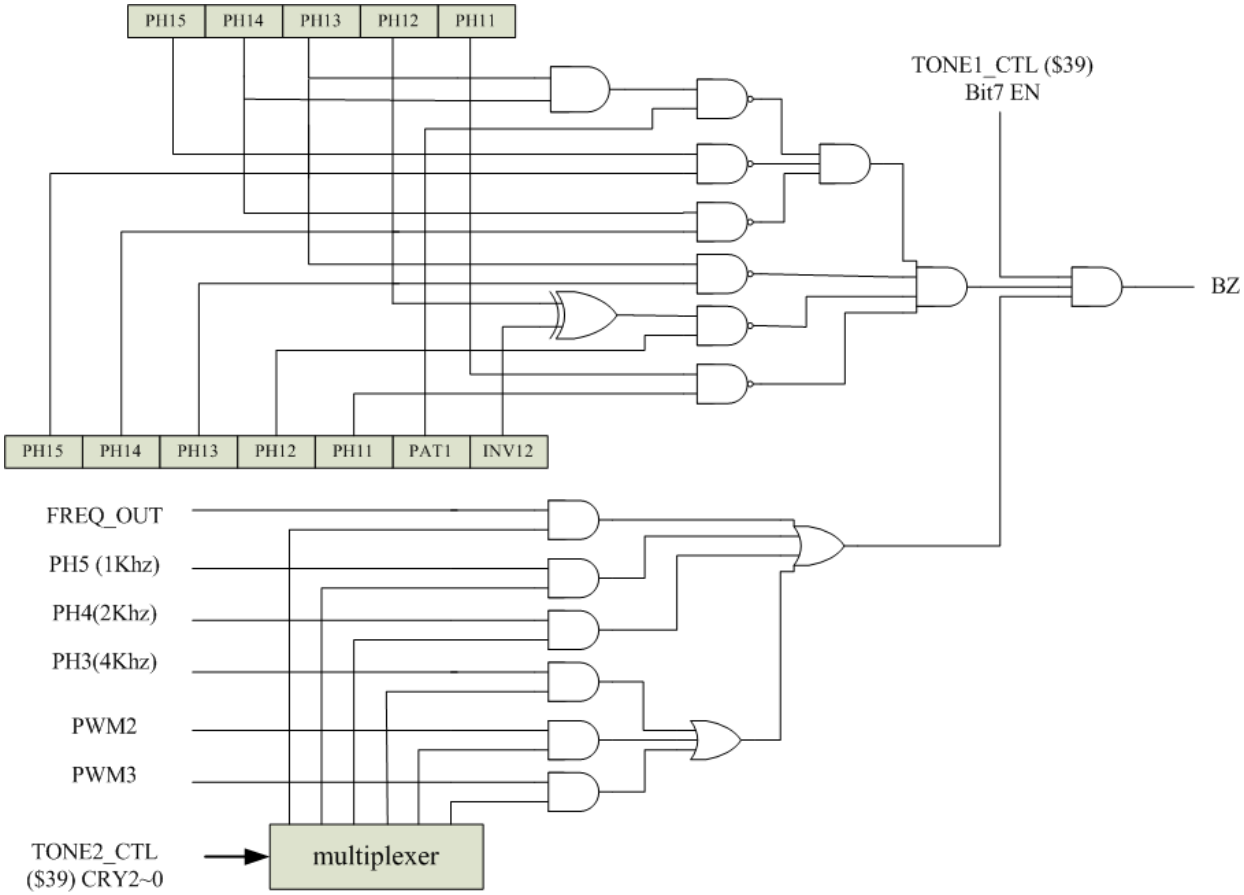


图.3.3.8 占空比设置定时图

图.3.3.9 蜂鸣器输出示例

```
            INC     'MK9A50P.inc'   ;; 蜂鸣器测试包括 TONE & 六载体
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x00            ;; 清除 TM2
            MOVAM   IRQF
            INC     RAM_*0,m        ;; 语音载体控制
            BTSS    RAM_80,5
            LGOTO   DATA_LOW
DATA_INC    INC     TONE_CTL2,m     ;; 语音载体选择信号
            CLR     RAM_80
DATA_LOW    NOP
            IRETI

            ORG     100h
START:      CLR     STATUS
            MOVLA   02h
            MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
            BC      SYS_CTL,b1      ;; 快时钟开启
            NOP
```

```
BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
NOP
CLR     RAM_80
MOVLA   0x0             ;; PD 输出
MOVAM   PD_DIR
MOVLA   B'01011111'
MOVAM   PAD_CTL2
MOVLA   B'11000111'     ;; ph0 输入，T:3T
MOVAM   TM0_CTL
MOVLA   B'1100000'
TM0_LA
MOVLA   B'0001000'
MOVAM   TONE_CTL1
CLR     TONE_CTL2
NOP
MOVLA   B'11011001'     ;; PH4，TM2 PWM 模式
MOVAM   TM2_CTL!
MOVLA   B'11011001'     ;; PH5，TM3 PWM 模式
MOVAM   TM3_CTL1
MOVLA   B'11101111'
MOVAM   TM2_LA
MOVAM   B'1111000'
MOVAM   TM3_LA
NOP
MOVLA   B'00001000'
MOVAM   IRQM
CLR     IRQF
BS      IRQM_CTL,7
BS      TONE_CTL1,7
LGOTO   $
```

3.4. 定时器 2 & 3（TM2 & TM3）

此定时器为多功能定时器，可设置为独立 8 位定时器使用。第二操作模式可作为捕捉事件计数器使用，用于计算来看自 CAPT1A 及 CAPT1B 脚位的外部事件。他们可作为两个独立的 8 位计数器。所有功能可通过下列寄存器设置，示意图也如下：

3.4.1 定时器 2

TM2_CTL1($1Fh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_CTL1	EN	WR_CNT	BIT	MOD1	MOD0	EDGE	SUR1	SUR0

Bit	符号	描 述			
7	EN	TM2使能/禁止 0：禁止 1：使能			
6	WR_CNT	TM2_CNT将通过写入数据到TM2_LA设置 （定时器，捕捉，PWM & RFC模式） 0：禁止 1：使能			
5	BIT	BIT：16-bit/8-bit 控制			
		0	8-bit 模式		
		1	16-bit 模式，TM2+TM3		
4~3	MOD1~0	MOD1~0：TM2操作模式选择			
		0 0	定时器模式		
		0 1	捕捉模式		
		1 0	RFC模式		
		1 1	PWM模式		
2	EDGE	捕捉信号沿控制位 1：外部时钟从H→L转换时增量 0：外部时钟从L→H转换时增量			
1~0	SUR1~0	时钟源 **（8-bit PWM 模式，PWM 占空比时钟源来自于 PH0X2）**			
				PWM 模式，BIT=0	PWM 模式，BIT=1
			定时器，捕捉	周期	周期
		0 0	FCLK（高速时钟）	PH3	FCLK（高速时钟）
		0 1	PH0 X 2	PH4	PH0 X 2
		1 0	PH4	PH5	PH4
		1 1	PH_CLK	PH_CLK	PH_CLK

TM2_CTL2($20h):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM2_CTL2	ENC	CLR_CNT		CAPIN1/ RFC_T1	CAPIN0/ RFC_T0	INT_S	PWM_OS	OV

Bit	符号	描 述		
7	ENC	捕捉 & RFC模式：计数器自动清除（当溢出时） 0：自动清除计数器（硬件模式） 1：通过软件清除计数器		
6	CLR_CNT	C捕捉 & RFC模式：清除计数器（当ENC=1及在捕捉模式或RFC模式下工作） 0：不清除 1：清除计数器及自动清除CLR_CNT		
4~3	CAPIN1~0/ RFC_T1~0	8 信号源选择（仅在捕捉模式下工作） 9 IRQ 源选择（仅在 RFC 模式下工作）		
		模式	捕捉模式	RFC 模式
		00	CAPT1A 输入	TMR0 IRQ
		01	CAPT1B 输入	PH IRQ
		10	CAPT2A 输入	TMR3 IRQ
		11	CAPT2B 输入	PH9
2	INT_S	信号源选择（在捕捉或 RFC 模式下工作）		
		INT_S	捕捉模式	RFC 模式
		0	**捕捉 IRQ**	**No IRQ**
		1	**捕捉溢出 IRQ**	**RFC 溢出 IRQ**
1	PWM_OS	PWM_OS：PWM选择位输出阶段		
		0	初始输出阶段是 L，当定时器溢出将变为 H	
		1	初始输出阶段是 H，当定时器溢出将变为 L	
0	OV	溢出位（捕捉 & RFC 模式，读取后用户应清除此位） 0：无溢出 1：溢出		

TM2_LA ($21h): TM2 data (R/W)

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($22h): TM2 counter (R/W) (up counter)

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

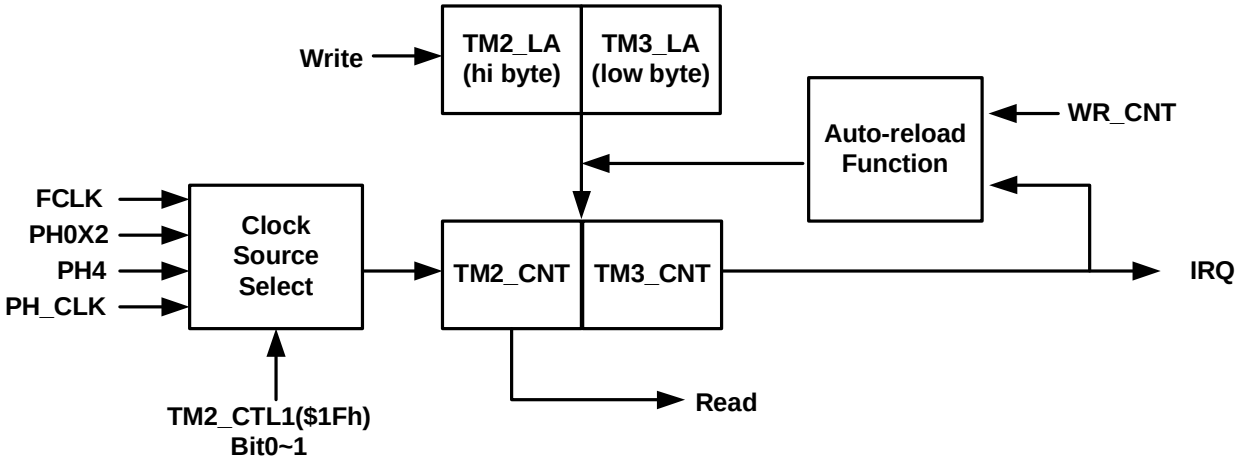


图.3.4.1 TM2+TM3 作为定时器的示意图（16-bit 模式）

图.3.4.1 TM2+TM3 16-bit 定时器模式示例

```
                INC     'MK9A50P.inc'   ;; TM2 中断
#DEFINE         RAM_80  80H
                ORG     00              ;;
                LGOTO   START

INT:            ORG     004
                MOVLA   0x7D            ;; 清除 TM2
                MOVAM   IRQF
                INC     PD_DAT,m        ;; 检测 TM2 IRQ
                MOVLA   0x010           ;; 改变 TM2 数据
                ADD     TM2_LA
                NOP
                IRETI

                ORG     100h
START:          CLR     STATUS
                MOVLA   02h
                MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
                BC      SYS_CTL,b1      ;; 快时钟开启
                NOP
                BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
                NOP
                CLR     RAM_80
                CLR     PA_DAT
                CLR     PD_DAT
                CLR     PA_DIR          ;; PA 输出
```

```
CLR     PAD_CTL1    ;; PD 作为 I/O 口工作
CLR     PD_DIR      ;; PD 输出
MOVLA   B’00110000’’    ;; PA3 = PWM1 输出
MOVAM   PAD_CTL2
MOVLA   B’01000001’     ;; 8-bit 定时器模式
MOVAM   TM2_CTL1
MOVLA   B’00000001’
MOVAM   TM2_CTL2
BS      TM2_CTL1,6  ;;  写 TM2_CNT 使能
BS      TM3_CTL1,6  ;;  写 TM3_CNT 使能
MOVLA   0x0         ;; TM2 上计数器 00 è FF
MOVAM   TM2_LA
MOVLA   0x80        ;; PWM 占空比计数器
MOVAM   TM3_LA
BC      TM2_CTL1,6
BC      TM3_CTL1,6
MOVLA   B’00000010’     ;;  设置 TM2 IRQ 掩膜
MOVAM   IRQM
CLR     IRQF
BS      IRQM_CTL,7
BS      TM2_CTL1,7
LGOTO   $
```

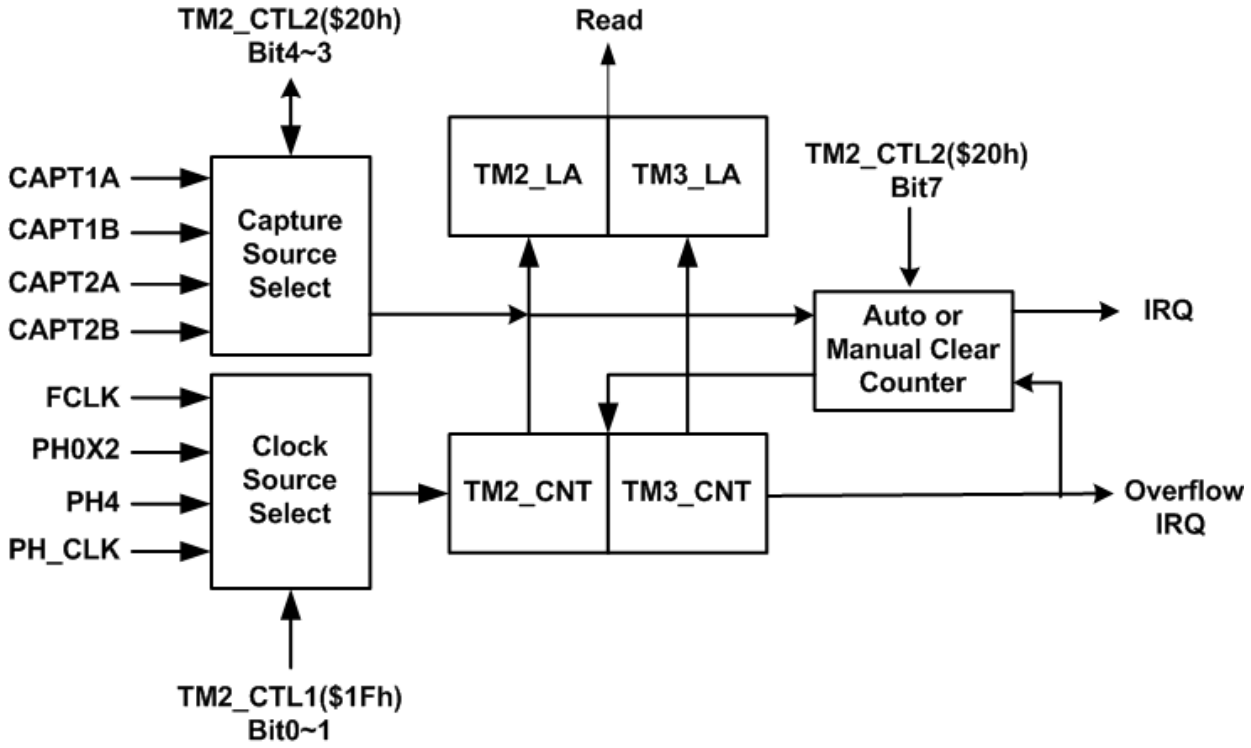


图.3.4.2 TM2+TM3 作为捕捉的示意民图（16-bit 模式）

图.3.4.2 TM2+TM3 16-bit捕捉模式示例

```
            INC     'MK9A50P.inc'   ;; TM2 中断
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x7D            ;; 清除 TM2
            MOVAM   IRQF
            MOV     TM2_LA          ;; 检测捕捉高字节数据
            MOVAM   PD_DAT
            MOV     TM3_LA          ;; 检测捕捉低字节数据
            MOVAM   PC_DAT
            IRETI

            ORG     100h
START:      CLR     STATUS
            MOVLA   02h
            MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
            BC      SYS_CTL,b1      ;; 快时钟开启
            NOP
            BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
            NOP
            CLR     RAM_80
            CLR     PA_DAT
```

```
        CLR     PD_DAT
        MOVLA   0xFF
        MOVAM   PA_DIR          ;; PA 输入
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        CLR     PC_DIR          ;; PC 输出
        MOVLA   B'00101001'     ;; 16-bit 捕捉，计数器 ph0x2
        MOVAM   TM2_CTL1
        BC      TM2_CTL2,3      ;; 捕捉输入 = PA3
        BS      TM2_CTL2,3      ;; 捕捉输入 = PA6
        MOVLA   B'00000010'     ;; 设置 TM2 IRQ 掩膜
        MOVAM   IRQM
        CLR     IRQF
        BS      IRQM_CTL,7
        BS      TM2_CTL1,7
        LGOTO   $
```

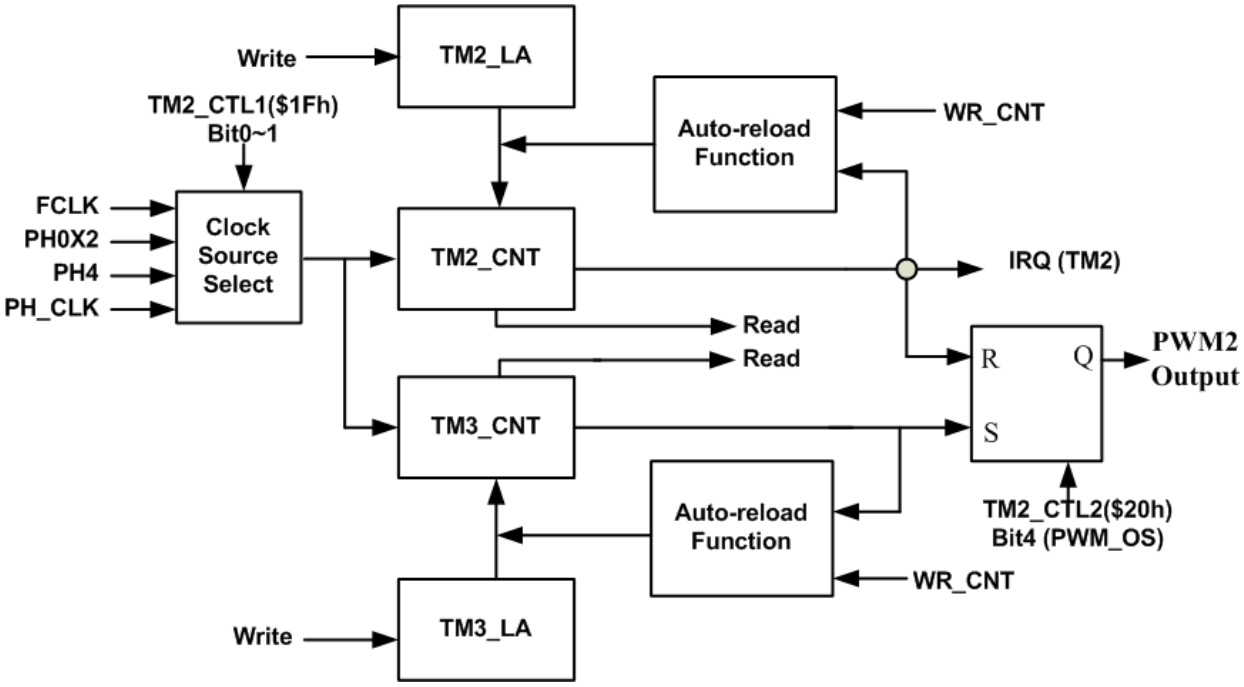


图.3.4.3 TM2+TM3 作为 PWM 的示意图（BIT=1）

图.3.4.3 PWM2（TM2+TM3 PWM）输出示例

```
                INC     'MK9A50P.inc'   ;; 蜂鸣器测试包括 TONE & 六载体
#DEFINE         RAM_80  80H
                ORG     00              ;;
                LGOTO   START

INT:            ORG     004
```

```
        MOVLA   0x7D            ;; 清除 TM2
        MOVAM   IRQF
        INC     PD_DAT,m        ;; 检测 TM2 IRQ
        MOVLA   0x010           ;; 改变 PWM 周期
        ADD     TM2_LA
        MOVLA   0x010           ;; 改变 PWM 占空比
        ADD     TM3_LA
        NOP
        IRETI

        ORG     100h
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;; 快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
        NOP
        CLR     RAM_80
        CLR     PA_DAT
        CLR     PD_DAT
        CLR     PA_DIR          ;; PA 输出
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        MOVLA   B'00110000''    ;; PA3 = PWM2 输出
        MOVAM   PAD_CTL2
        MOVLA   B'00111000'     ;; PWM 模式，TM2 作为周期工作，TM3 作为占空比
                                   工作
        MOVAM   TM2_CTL1
        MOVLA   B'00000001'
        MOVAM   TM2_CTL2
        BS      TM2_CTL1,6      ;; 写 TM2_CNT 使能
        BS      TM3_CTL1,6      ;; 写 TM3_CNT 使能
        MOVLA   0x20            ;; PWM 周期计数器
        MOVAM   TM2_LA
        MOVLA   0x80            ;; PWM 占空比计数器
        MOVAM   TM3_LA
        BC      TM2_CTL1,6
        BC      TM3_CTL1,6
        MOVLA   B'00000010'     ;; 设置 TM2 IRQ 掩膜
        MOVAM   IRQM
        CLR     IRQF
```

```
        BS      IRQM_CTL,7
        BS      TM2_CTL1,7
        LGOTO   $
```

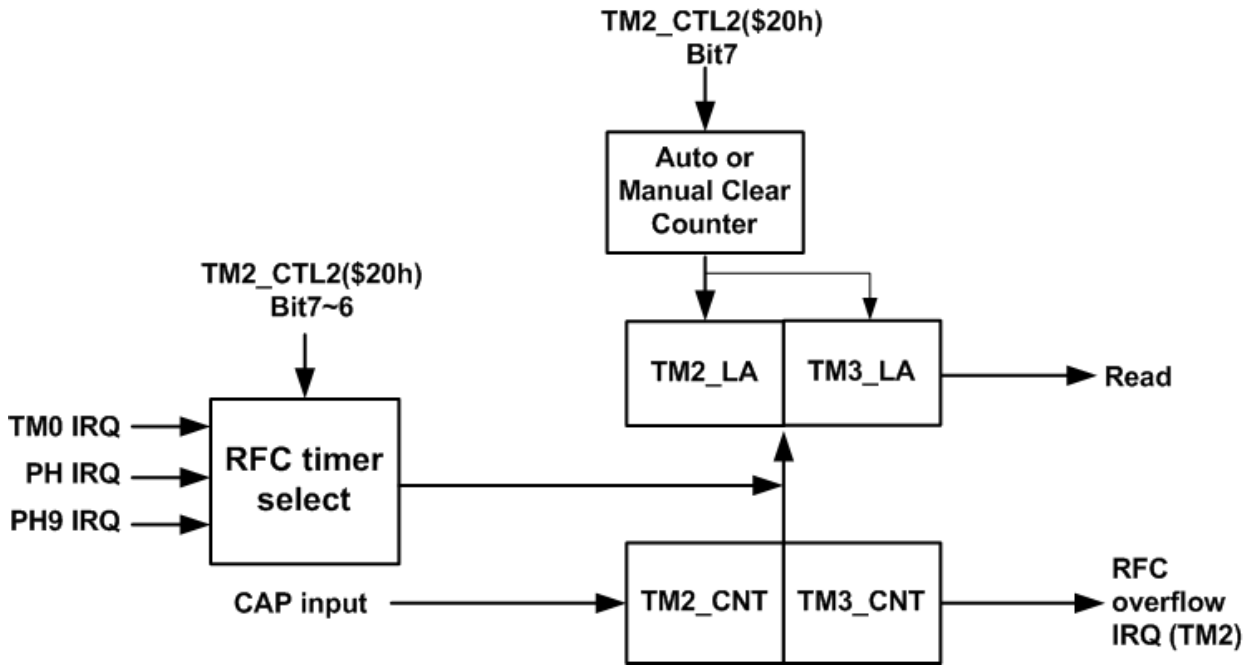


图.3.4.4 TM2+TM3 作为 RFC 的示意大图（16-bit 模式）

图.3.4.4 TM2+TM3 16-bit RFC 模式示例

```
        INC     'MK9A50P.inc'   ;; TM2 中断
#DEFINE RAM_80  80H
        ORG     00              ;;
        LGOTO   START

INT:    ORG     004
        MOVLA   0x7E            ;; 清除 TM0
        MOVAM   IRQF
        MOV     TM2_LA          ;; 检测 RFC 高字节数据
        MOVAM   PD_DAT
        MOV     TM3_LA          ;; 检测 RFC 低字节数据
        MOVAM   PC_DAT
        IRETI

        ORG     100h
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;; 快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
        NOP
```

```
CLR     RAM_80
CLR     PA_DAT
CLR     PD_DAT
CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
CLR     PD_DIR          ;; PD 输出
CLR     PC_DIR          ;; PC 输出

MOVLA   0xFF
MOVAM   PA_DIR          ;; PA 输入
MOVLA   B'11101111'     ;; RFC，BZ & BZM 输出
MOVAM   PAD_CTL2
BS      PAD_CTL3,0      ;; RREF ON
;BS     PAD_CTL3,1      ;; SEN0 ON
;BS     PAD_CTL3,2      ;; SEN1 ON
MOVAM
MOVLA   B'00111000'     ;; 16-bit RFC
MOVAM   TM2_CTL1        ;; RFC IRQ 来自 TM0
CLR     TM2_CTL2
MOVLA   B'01000100'     ;; RFC，PH0X2，T/2:T/2
MOVAM   TM0_CTL
MOVLA   B'00111111'
MOVAM   TM0_LA
BC      TM0_CTL,6

MOVLA   B'00000001'     ;; 设置 TM0 IRQ 掩膜
MOVAM   IRQM
BS      IRQM_CTL,7
BS      TM2_CTL1,7
BS      TM0_CTL1,7
        CLR             IRQF
LGOTO   $
```

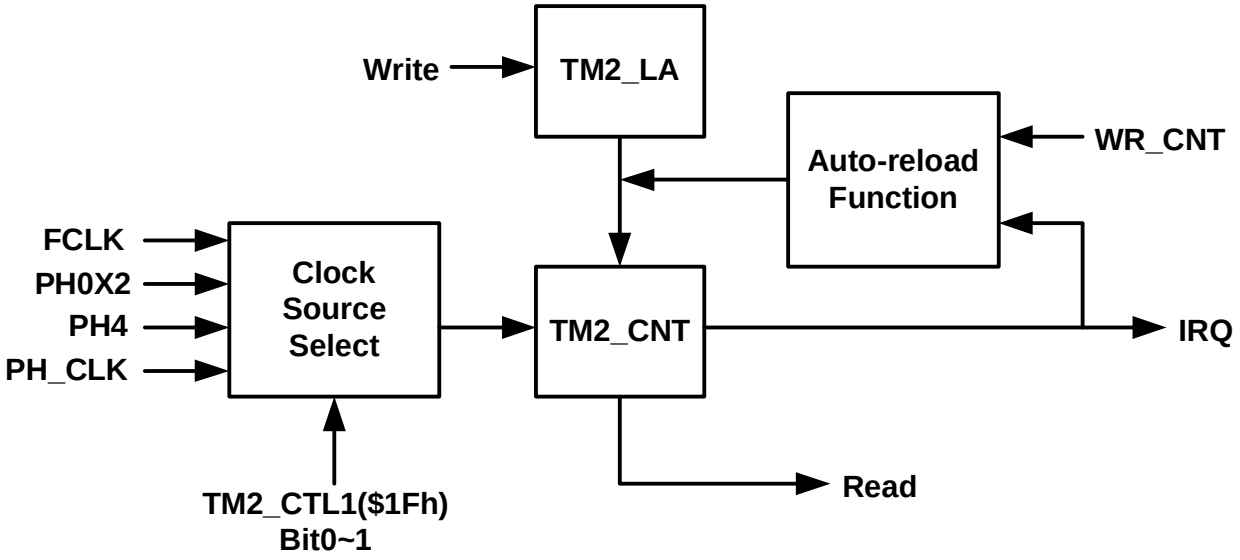


图.3.4.5 TM2 作为定时器工作的示意图（8-bit 模式）

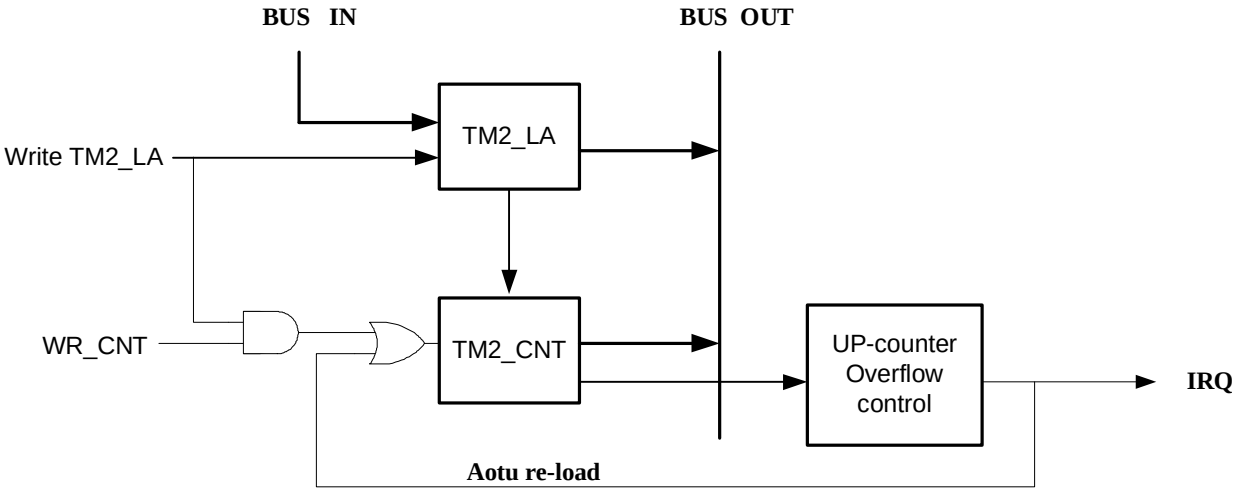


图.3.4.6 TM2 作为定时器工作的示意图（8-bit 模式）

图.3.4.5 TM2 8-bit定时器模式示例

```
            INC     'MK9A50P.inc'   ;; TM2 8-bit 定时器模式
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x7D            ;; 清除 TM2
            MOVAM   IRQF
            INC     PD_DAT,m        ;; 检测 TM2 IRQ
            MOVLA   0x010           ;; 改变 TM2 数据
            ADD     TM2_LA
            NOP
            IRETI

            ORG     100h
```

```
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;;  快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟  = FCLK
        NOP
        CLR     RAM_80
        CLR     PA_DAT
        CLR     PD_DAT
        CLR     PA_DIR          ;; PA 输出
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        MOVLA   B'00110000''    ;; PA3 = PWM1 输出
        MOVAM   PAD_CTL2
        MOVLA   B'01000001'     ;; 8-bit 定时器模式
        MOVAM   TM2_CTL1
        MOVLA   B'00000001'
        MOVAM   TM2_CTL2
        BS      TM2_CTL1,6      ;;  写 TM2_CNT 使能
        MOVLA   0x0             ;; TM2 上计数器 00 è FF
        MOVAM   TM2_LA
        BC      TM2_CTL1,6
        MOVLA   B'00000010'     ;;  设置 TM2 IRQ 掩膜
        MOVAM   IRQM
        CLR     IRQF
        BS      IRQM_CTL,7
        BS      TM2_CTL1,7
        LGOTO   $
```

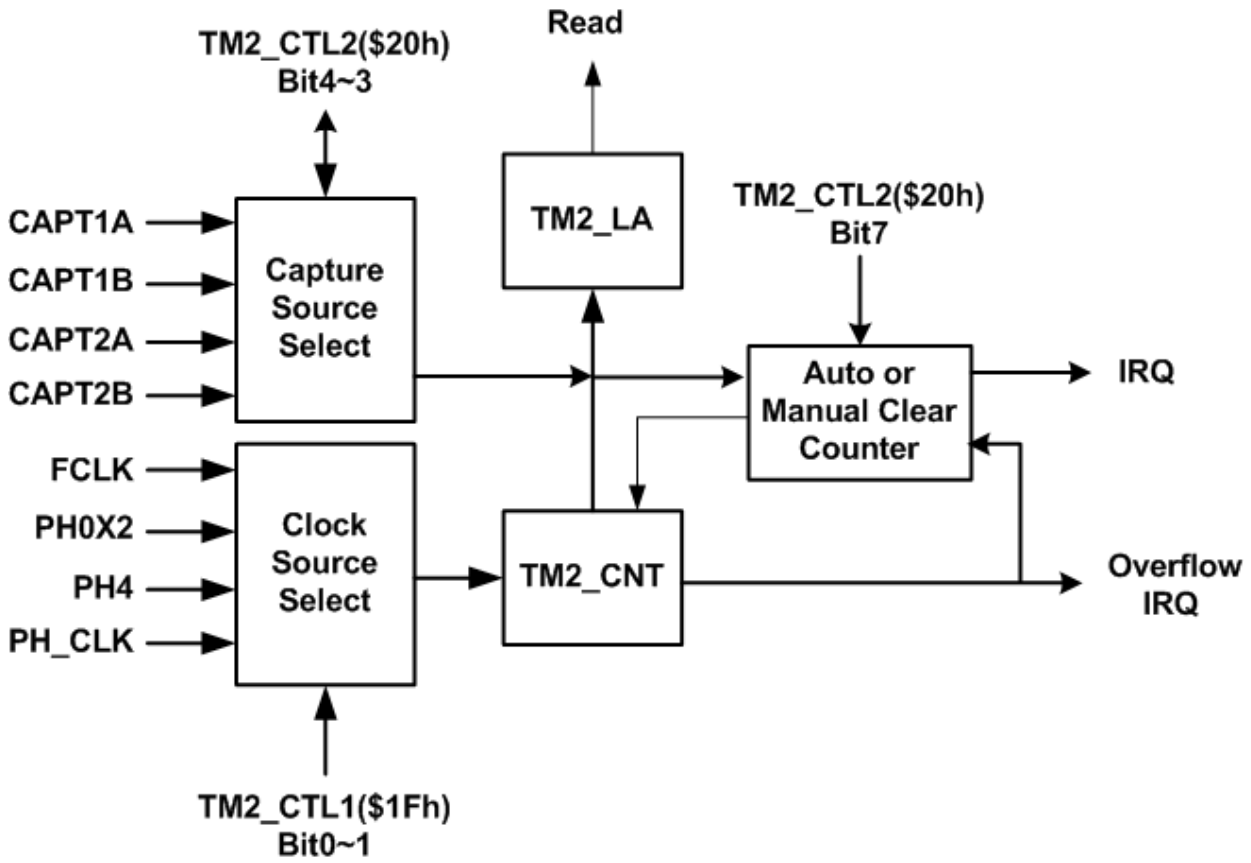


图.3.4.7 TM2 作为捕捉的示意图（8-bit 模式）

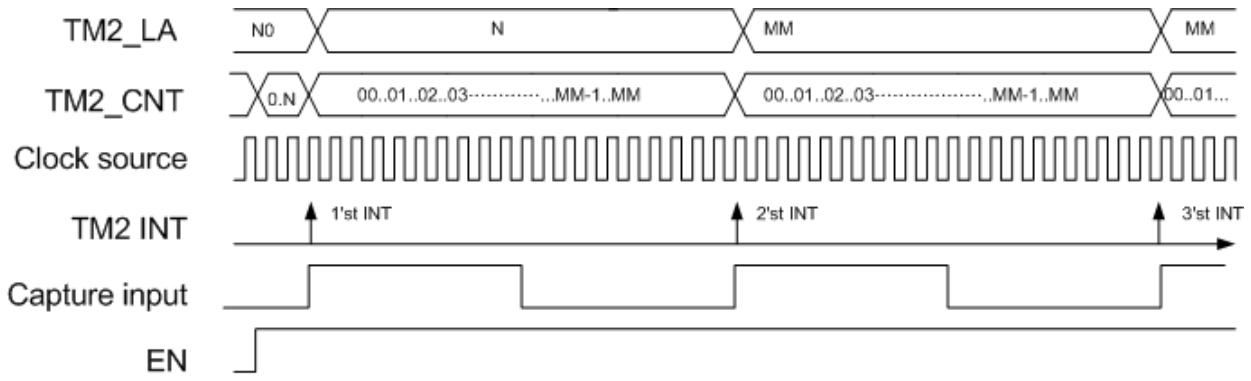


图.3.4.8 TM2 作为捕捉的示意图（循环，EDGE=0）（8-bit 模式）

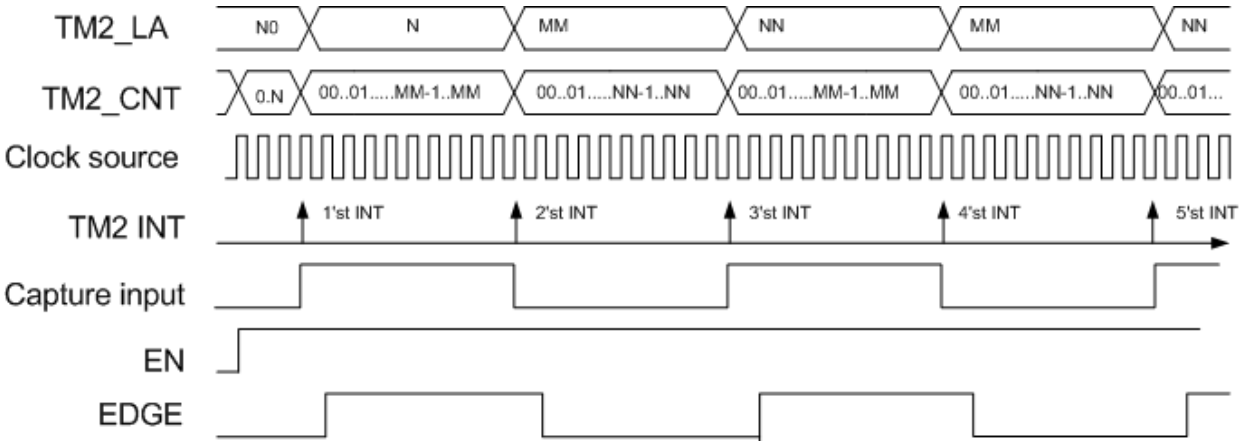


图.3.4.9 TM2 作为捕捉的示意图（脉冲）（8-bit 模式）

图.3.4.7 TM2 8-bit捕捉模式示例

```
            INC     'MK9A50P.inc'   ;; TM2 8-bit 捕捉模式
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x7D            ;; 清除 TM2
            MOVAM   IRQF
            MOV     TM2_LA          ;; 检测 RFC 高字节数据
            MOVAM   PD_DAT
            IRETI

            ORG     100h
START:      CLR     STATUS
            MOVLA   02h
            MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
            BC      SYS_CTL,b1      ;; 快时钟开启
            NOP
            BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
            NOP
            CLR     RAM_80
            CLR     PA_DAT
            CLR     PD_DAT
            MOVLA   0xFF
            MOVAM   PA_DIR          ;; PA 输入
            CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
            CLR     PD_DIR          ;; PD 输出
            CLR     PC_DIR          ;; PC 输出
            MOVLA   B'00001001'     ;; 8-bit 捕捉，计数器 ph0x2
            MOVAM   TM2_CTL1
```

```
BC      TM2_CTL2,3      ;; 捕捉输入 = PA3
;; BS   TM2_CTL2,3      ;; 捕捉输入 = PA6
MOVLA   B'00000010'     ;; 设置 TM2 IRQ 掩膜
MOVAM   IRQM
CLR     IRQF
BS      IRQM_CTL,7
BS      TM2_CTL1,7
LGOTO   $
```

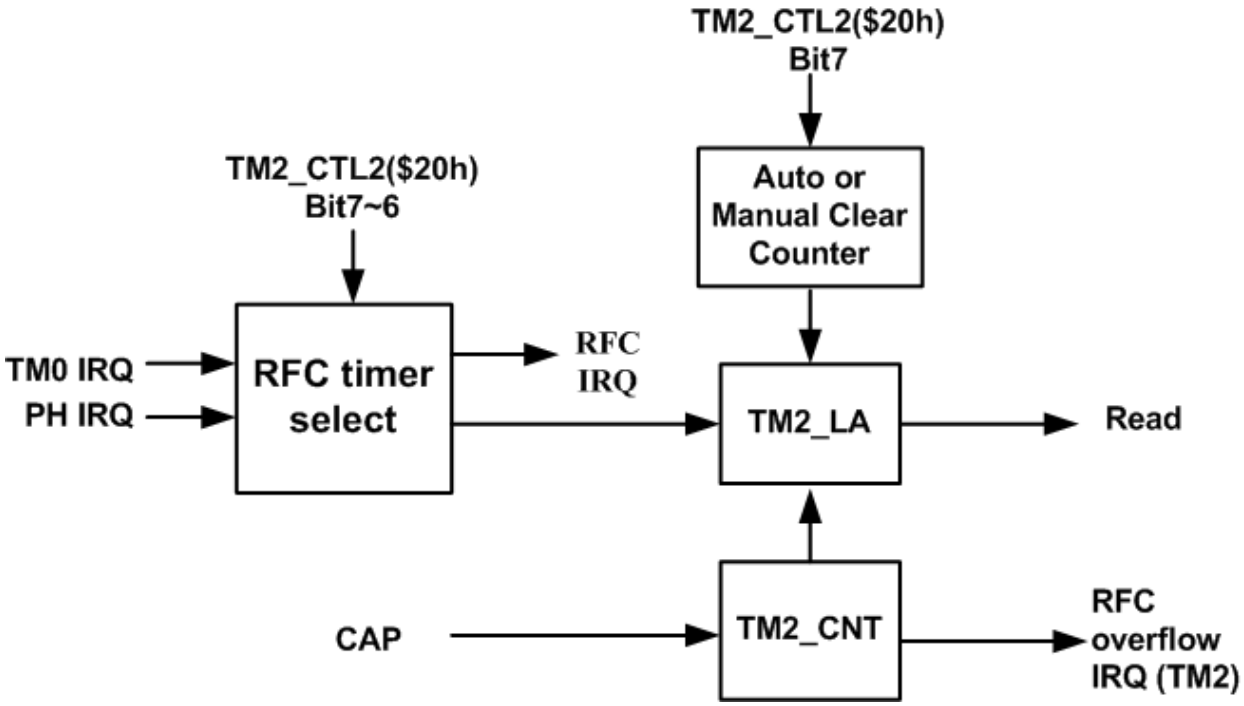


图.3.4.10 TM2 作为 RFC 的示意图（8-bit 模式）

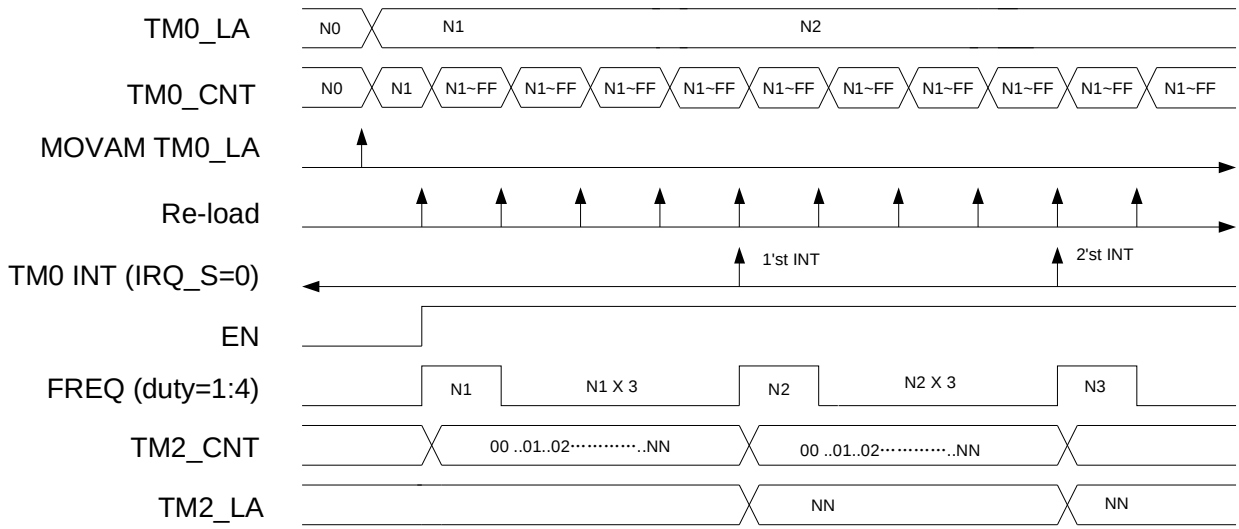


图.3.4.11 TM2 作为 RFC 的示意图（软件模式，ENC=1）（8-bit 模式）

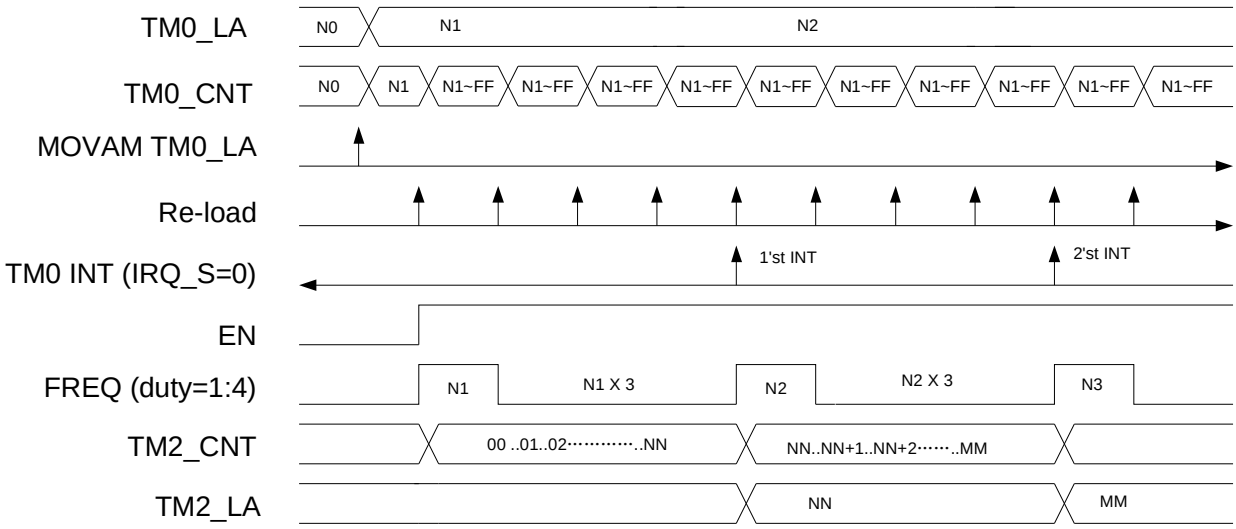


图.3.4.12 TM2 作为 RFC 的示意图（硬件模式，ENC=0）（8-bit 模式）

图.3.4.10 TM2 8-bit RFC模式示例

```
            INC     'MK9A50P.inc'   ;; TM2 8-bit RFC 模式
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x7E            ;; 清除 TM0
            MOVAM   IRQF
            MOV     TM2_LA          ;; 检测 RFC 数据
            MOVAM   PD_DAT
            IRETI

            ORG     100h
START:      CLR     STATUS
            MOVLA   02h
            MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
            BC      SYS_CTL,b1      ;; 快时钟开启
            NOP
            BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
            NOP
            CLR     RAM_80
            CLR     PA_DAT
            CLR     PD_DAT
            CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
            CLR     PD_DIR          ;; PD 输出
            CLR     PC_DIR          ;; PC 输出

            MOVLA   0xFF
            MOVAM   PA_DIR          ;; PA 输入
```

```
MOVLA   B’11101111’     ;; RFC，BZ & BZM 输出
MOVAM   PAD_CTL2
BS      PAD_CTL3,0      ;; RREF ON
;BS     PAD_CTL3,1      ;; SEN0 ON
;BS     PAD_CTL3,2      ;; SEN1 ON
MOVAM
MOVLA   B’00011000’     ;; TM2 作为 8-bit RFC 工作
MOVAM   TM2_CTL1        ;; RFC IRQ 来自 TM0
CLR     TM2_CTL2
MOVLA   B’01000100’     ;; RFC，PH0X2，T/2:T/2
MOVAM   TM0_CTL
MOVLA   B’00111111’
MOVAM   TM0_LA
BC      TM0_CTL,6

MOVLA   B’00000001’     ;; 设置 TM0 IRQ 掩膜
MOVAM   IRQM
BS      IRQM_CTL,7
BS      TM2_CTL1,7
BS      TM0_CTL1,7
        CLR             IRQF
LGOTO   $
```

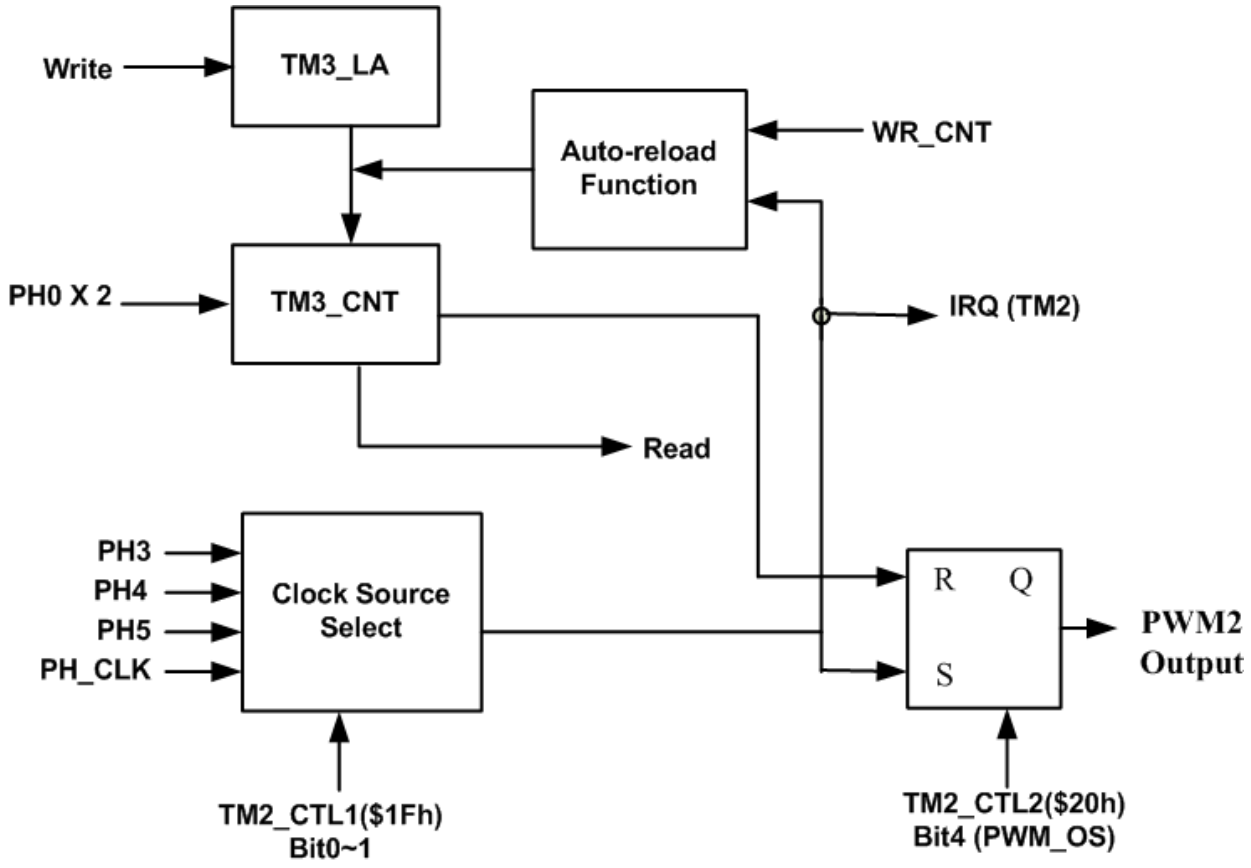


图.3.4.13 TM2 作为 PWM 的示意图（8-bit 模式）

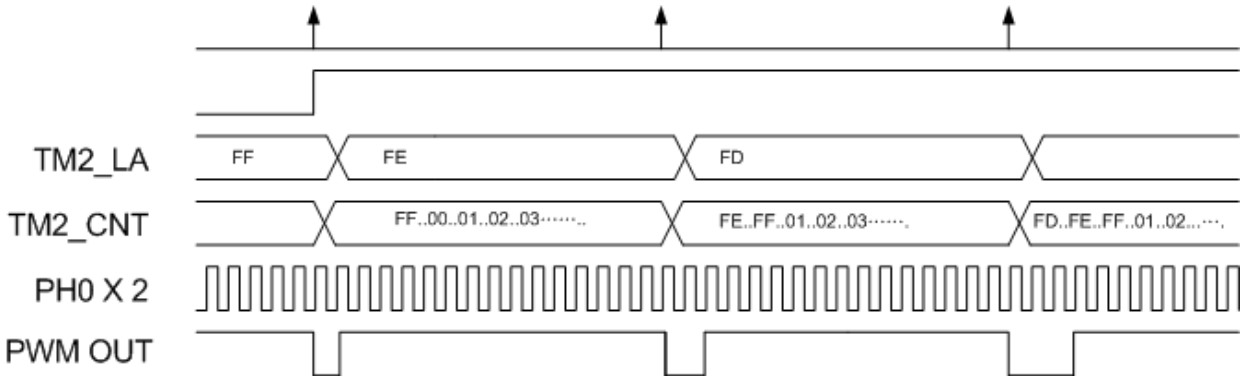


图.3.4.14 PWM 波形（PWM_OS=0，Period=PH3）（8-bit 模式）

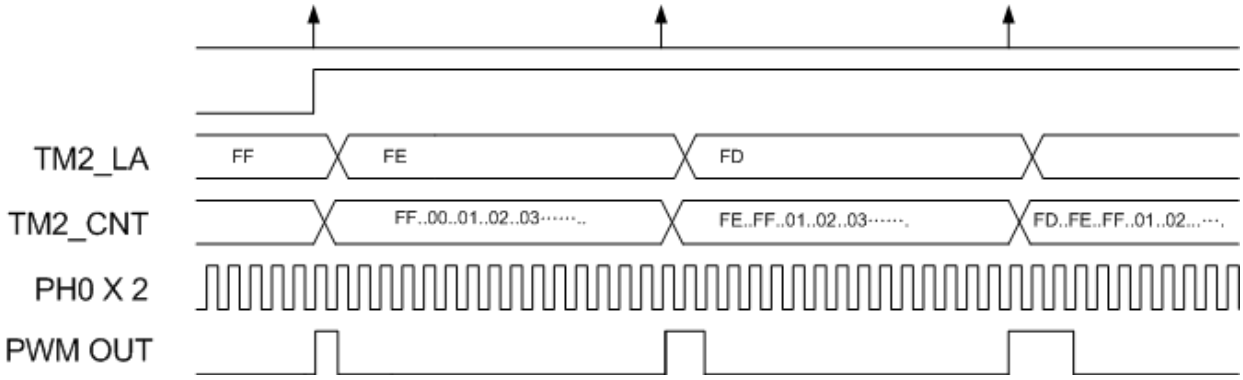


图.3.4.15　PWM 波形（PWM_OS=0，Period=PH3）（8-bit 模式）

TM2_LA	PWM2 (L : Hi)	TM2_LA	PWM1 (L : Hi)
FF	1 : 15	F7	9:7
FE	2 : 14	F6	10:6
FD	3 : 13	F5	11:5
FC	4 : 12	F4	12:4
FB	5 : 11	F3	13:3
FA	6 : 10	F2	14:2
F9	7 : 9	F1	15:1
F8	8 : 8	**TM2_LA < F1**	总是“0”

图.3.4.16 PWM 表格 1（PWM_OS=0，Period=PH3）（8-bit 模式）

TM2_LA	PWM2 (Hi : L)	TM2_LA	PWM1 (Hi : L)
FF	1 : 15	F7	9:7
FE	2 : 14	F6	10:6
FD	3 : 13	F5	11:5
FC	4 : 12	F4	12:4
FB	5 : 11	F3	13:3
FA	6 : 10	F2	14:2
F9	7 : 9	F1	15:1
F8	8 : 8	**TM2_LA < F1**	总是“1”

图.3.4.17 PWM 表格 2（PWM_OS=1，Period=PH3）（8-bit 模式）

TM2_LA	PWM2 (L : Hi)	TM2_LA	PWM2 (L : Hi)
FF	1 : 31	F7	9 : 23
FE	2 : 30	F6	10 : 22
FD	3 : 29	F5	11 : 21
FC	4 : 28	F4	12 : 20
FB	5 : 27	F3	13 : 19
FA	6 : 26	F2	14 : 18
F9	7 : 25	F1	15 : 17
F8	8 : 24	**TM2_LA < E1**	总是“0”

图.3.4.18 PWM 表格 3（PWM_OS=0，Period=PH4）（8-bit 模式）

TM2_LA	PWM2 (Hi : L)	TM2_LA	PWM2 (Hi : L)
FF	1 : 31	F7	9 : 23
FE	2 : 30	F6	10 : 22
FD	3 : 29	F5	11 : 21
FC	4 : 28	F4	12 : 20
FB	5 : 27	F3	13 : 19
FA	6 : 26	F2	14 : 18
F9	7 : 25	F1	15 : 17
F8	8 : 24	**TM2_LA < E1**	总是“1”

图.3.4.19 PWM 表格 4（PWM_OS=1，Period=PH4）（8-bit 模式）

TM2_LA	PWM2 (L : Hi)	TM2_LA	PWM2 (L : Hi)
FF	1 : 63	F7	9 : 55
FE	2 : 62	F6	10 : 54
FD	3 : 61	F5	11 : 53
FC	4 : 60	F4	12 : 52
FB	5 : 59	F3	13 : 51
FA	6 : 58	F2	14 : 50
F9	7 : 57	F1	15 : 49
F8	8 : 56	**TM2_LA < C1**	总是“0”

图.3.4.20 PWM 表格 5（PWM_OS=0，Period=PH5）（8-bit 模式）

TM2_LA	PWM2 (Hi : L)	TM2_LA	PWM2 (Hi : L)
FF	1 : 63	F7	9 : 55
FE	2 : 62	F6	10 : 54
FD	3 : 61	F5	11 : 53
FC	4 : 60	F4	12 : 52
FB	5 : 59	F3	13 : 51
FA	6 : 58	F2	14 : 50
F9	7 : 57	F1	15 : 49
F8	8 : 56	**TM2_LA < C1**	总是“1”

图.3.4.21 PWM 表格 6（PWM_OS=1，Period=PH4）（8-bit 模式）

TM2_LA	PWM2 (L : Hi)	TM2_LA	PWM2 (L : Hi)
FF	1 : 255	F7	9 : 247
FE	2 : 254	F6	10 : 246
FD	3 : 253	F5	11 : 245
FC	4 : 252	F4	12 : 244
FB	5 : 251	F3	13 : 243
FA	6 : 250	F2	14 : 242
F9	7 : 249	MM	**(FF-MM+1) : MM**
F8	8 : 248	0	总是“0”

图.3.4.22 PWM 表格 7（PWM_OS=0，Period=PH7）（8-bit 模式）

TM2_LA	PWM2 (Hi : L)	TM2_LA	PWM2 (Hi : L)
FF	1 : 255	F7	9 : 247
FE	2 : 254	F6	10 : 246
FD	3 : 253	F5	11 : 245
FC	4 : 252	F4	12 : 244
FB	5 : 251	F3	13 : 243
FA	6 : 250	F2	14 : 242
F9	7 : 249	MM	**(FF-MM+1) : MM**
F8	8 : 248	0	总是“1”

图.3.4.23 PWM 表格 8（PWM_OS=1，Period=PH7）（8-bit 模式）

图.3.4.13 PWM2 输出（TM2 8-bit PWM 输出）示例

```
            INC     'MK9A50P.inc'   ;; TM2 PWM 模式
                                    ;; Period=PH7，占空比来自 TM2
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START
```

```
INT:    ORG     004
        MOVLA   0x7D            ;; 清除 TM2
        MOVAM   IRQF
        INC     PD_DAT,m        ;; 检测 TM2 IRQ
        MOVLA   0x01
        ADD     TM2_LA          ;; 改变 PWM 占空比
        NOP
        IRETI

        ORG     100h
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;; 快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
        NOP
        CLR     RAM_80
        CLR     PA_DIR          ;; PA 输出
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        MOVLA   B’00110000’’    ;; PA3 = PWM2 输出
        MOVAM   PAD_CTL2
        MOVLA   B’01011000’     ;; PWm 模式，PWM 占空比=ph3
        MOVAM   TM2_CTL1
        MOVLA   B’00000001’
        MOVAM   TM2_CTL2
        MOVLA   0xE0            ;; PWM 占空比计数器
        MOVAM   TM2_LA
        BC      TM2_CTL1,6
        NOP
        MOVLA   B’00000010’     ;; 设置 TM2 IRQ 掩膜
        MOVAM   IRQM
        CLR     IRQF
        BS      IRQM_CTL,7
        BS      TM2_CTL1,7
        LGOTO   $
```

3.4.2 定时器 3

TM3_CTL1($23h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM3_CTL1	EN	WR_CNT	--	MOD1	MOD0	EDGE	SUR1	SUR0

Bit	符号	描 述			
7	EN	TM3使能/禁止 0：禁止 1：使能			
6	WR_CNT	TM3_CNT将通过写入数据到TM3_LA被设置 （定时器，捕捉，PWM & RFC模式） 0：禁止 1：使能			
4~3	MOD1~0	MOD1~0：TM3操作模式选择			
		0 0	定时器模式		
		0 1	捕捉模式		
		1 0	RFC模式		
		1 1	PWM模式（TM3来自PH0X2输入）		
2	EDGE	捕捉信号沿控制位 1：外部时钟从H→L转换时增量 0：外部时钟从L→H转换时增量			
1~0	SUR1~0	时钟源 **（8-bit PWM 模式，PWM 占空比时钟源来自 PH0X2）**			
				PWM 模式，BIT=0	PWM 模式，BIT=1
			定时器，捕捉	周期	占空比
		0 0	FCLK（高速时钟）	PH3	FCLK（高速时钟）
		0 1	PH0 X 2	PH4	PH0 X 2
		1 0	PH4	PH5	PH4
		1 1	PH_CLK	PH_CLK	PH_CLK

TM3_CTL2($24h):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM3_CTL2	ENC	CLR_CNT		CAPIN1/ RFC_T1	CAPIN0/ RFC_T0	INT_S	PWM_OS	OV

<table>
<tr><th>Bit</th><th>符号</th><th colspan="3">描 述</th></tr>
<tr><td>7</td><td>ENC</td><td colspan="3">捕捉 & RFC模式：计数器自动清除（当溢出时）
0：自动清除计数器（硬件模式）
1：通过软件清除计数器</td></tr>
<tr><td>6</td><td>CLR_CNT</td><td colspan="3">捕捉 & RFC模式：清除计数器（当ENC=1及在捕捉模式或RFC模式下工作时）
0：不清除
1：清除计数器及自动清除CLR_CNT</td></tr>
<tr><td rowspan="6">4~3</td><td rowspan="6">CAPIN1~0/
RFC_T1~0</td><td colspan="3">1. 信号源选择（仅在捕捉模式下工作）
2. IRQ 源选择（仅在 RFC 模式下工作）</td></tr>
<tr><td>模式</td><td>捕捉模式</td><td>RFC 模式</td></tr>
<tr><td>00</td><td>CAPT1A 输入</td><td>TMR0 IRQ</td></tr>
<tr><td>01</td><td>CAPT1B 输入</td><td>PH IRQ</td></tr>
<tr><td>10</td><td>CAPT2A 输入</td><td>TMR2 IRQ</td></tr>
<tr><td>11</td><td>CAPT2B 输入</td><td>PH9</td></tr>
<tr><td rowspan="4">2</td><td rowspan="4">INT_S</td><td colspan="3">信号源选择（在捕捉及 RFC 模式下工作）</td></tr>
<tr><td>INT_S</td><td>捕捉模式</td><td>RFC 模式</td></tr>
<tr><td>0</td><td>捕捉 IRQ</td><td>No IRQ</td></tr>
<tr><td>1</td><td>捕捉溢出 IRQ</td><td>RFC 溢出 IRQ</td></tr>
<tr><td rowspan="3">1</td><td rowspan="3">PWM_OS</td><td colspan="3">PWM_OS：PWM选择位的输出阶段</td></tr>
<tr><td>0</td><td colspan="2">初始输出状态是 H，当定时器溢出将改变为 L</td></tr>
<tr><td>1</td><td colspan="2">初始输出状态是 L，当定时器溢出将改变为 H</td></tr>
<tr><td>0</td><td>OV</td><td colspan="3">溢出位（仅捕捉 & RFC 模式，读取后用户应清除此位）
0：无溢出
1：溢出</td></tr>
</table>

TM3_LA ($25h): TM3 数据（R/W）

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($26h): TM3 计数器（R/W）（上计数器）

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

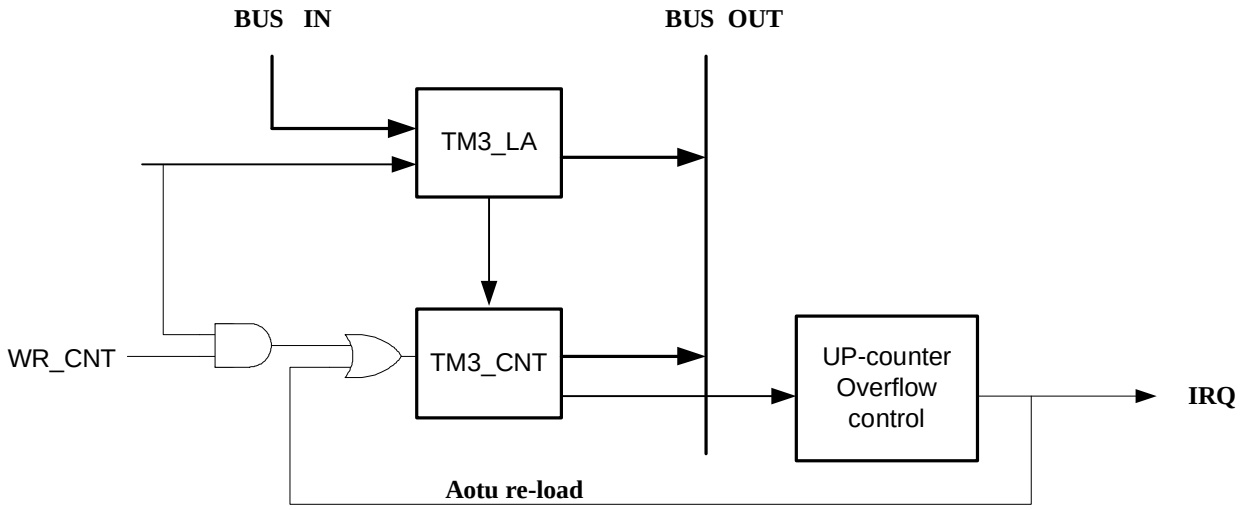


图.3.4.24 TM3 自动重载功能示意图

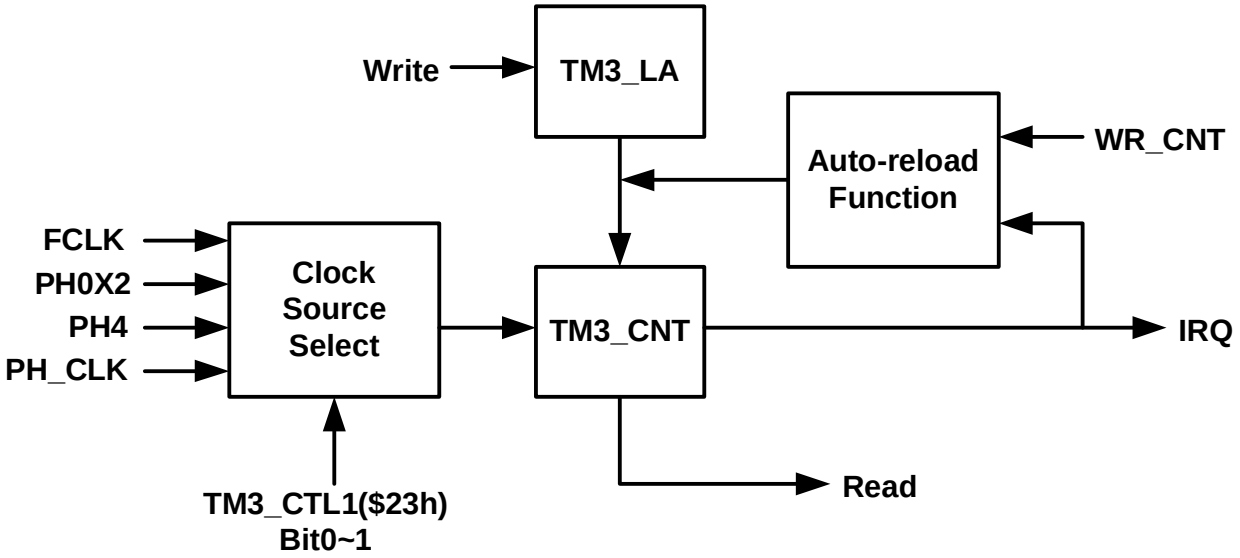


图.3.4.25 TM3 作为定时器的示意图

图.3.4.25 TM3 8-bit定时器模式示例

```
                INC       'MK9A50P.inc'     ;; TM3 8-bit 定时器模式
#DEFINE         RAM_80    80H
                ORG       00                ;;
                LGOTO     START

INT:            ORG       004
                MOVLA     0x7B              ;; 清除 TM3
                MOVAM     IRQF
                INC       PD_DAT,m          ;; 检测 TM3 IRQ
                MOVLA     0x010             ;; 改变 TM3 数据
                ADD       TM3_LA
                NOP
                IRETI

                ORG       100h
```

```
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;; 快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
        NOP
        CLR     RAM_80
        CLR     PA_DAT
        CLR     PD_DAT
        CLR     PA_DIR          ;; PA 输出
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        MOVLA   B'01000001'     ;; 8-bit 定时器模式
        MOVAM   TM3_CTL1
        MOVLA   B'00000001'
        MOVAM   TM3_CTL2
        BS      TM3_CTL1,6      ;; 写 TM3_CNT 使能
        MOVLA   0x0             ;; TM3 上计数器，00 è FF
        MOVAM   TM3_LA
        BC      TM3_CTL1,6
        MOVLA   B'00000100'     ;; 设置 TM3 IRQ 掩膜
        MOVAM   IRQM
        CLR     IRQF
        BS      IRQM_CTL,7
        BS      TM3_CTL1,7
        LGOTO   $
```

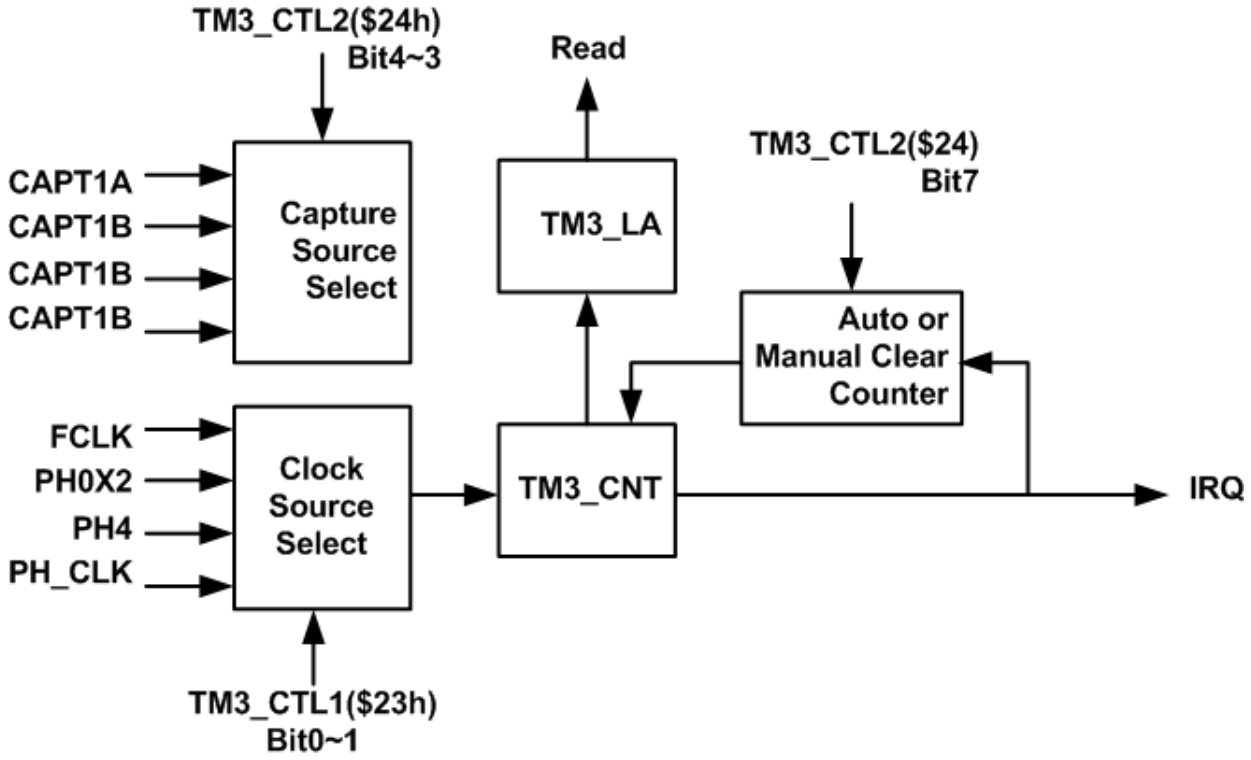


图.3.4.26 TM3 作为捕捉的示意图

图.3.4.26 TM3 8-bit 捕捉模式示例

```
            INC     'MK9A50P.inc'   ;; TM2 捕捉
#DEFINE     RAM_80  80H
            ORG     00              ;;
            LGOTO   START

INT:        ORG     004
            MOVLA   0x7B            ;; 清除 TM3
            MOVAM   IRQF
            MOV     TM3_LA          ;; 检测 RFC 高字节数据
            MOVAM   PD_DAT
            IRETI

            ORG     100h
START:      CLR     STATUS
            MOVLA   02h
            MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
            BC      SYS_CTL,b1      ;; 快时钟开启
            NOP
            BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
            NOP
            CLR     RAM_80
            CLR     PA_DAT
            CLR     PD_DAT
```

```
MOVLA     0xFF
MOVAM     PA_DIR          ;; PA 输入
CLR       PAD_CTL1        ;; PD 作为 I/O 口工作
CLR       PD_DIR          ;; PD 输出
CLR       PC_DIR          ;; PC 输出
BC        TM2_CTL1,5      ;; 8-bit 模式控制
MOVLA     B'00001001'     ;; 8-bit 捕捉，计数器 ph0x2
MOVAM     TM3_CTL1
BC        TM3_CTL2,3      ;;  捕捉输入  = PA3
;; BS     TM3_CTL2,3      ;;  捕捉输入  = PA6
MOVLA     B'00000100'     ;;  设置 TM3 IRQ 掩膜
MOVAM     IRQM
CLR       IRQF
BS        IRQM_CTL,7
BS        TM3_CTL1,7
LGOTO     $
```

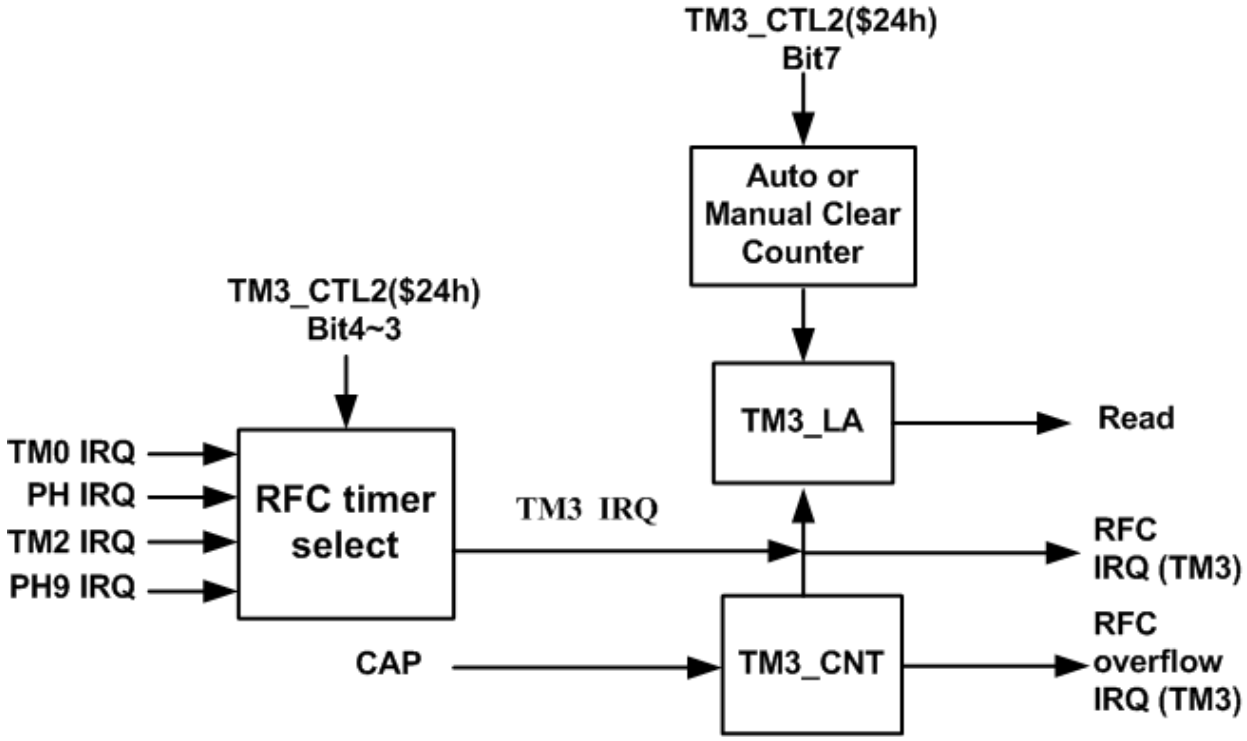


图.3.4.27 TM3 作为 RFC 的示意图

图.3.4.27 TM3 8-bit RFC模式示例

```
          INC       'MK9A50P.inc'   ;; TM3 RFC
#DEFINE   RAM_80    80H
          ORG       00              ;;
          LGOTO     START
```

```
INT:    ORG     004
        MOVLA   0x7E            ;; 清除 TM0
        MOVAM   IRQF
        MOV     TM3_LA          ;; 检测 RFC 数据
        MOVAM   PD_DAT
        IRETI

        ORG     100h
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;; 快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟 = FCLK
        NOP
        CLR     RAM_80
        CLR     PA_DAT
        CLR     PD_DAT
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        CLR     PC_DIR          ;; PC 输出

        MOVLA   0xFF
        MOVAM   PA_DIR          ;; PA 输入
        MOVLA   B'11101111'     ;; RFC，BZ & BZM 输出
        MOVAM   PAD_CTL2
        BS      PAD_CTL3,0      ;; RREF ON
        ;BS     PAD_CTL3,1      ;; SEN0 ON
        ;BS     PAD_CTL3,2      ;; SEN1 ON
        BC      TM2_CTL1,5      ;; 8-bit 模式控制
        MOVLA   B'00011000'     ;; TM2 作为 8-bit RFC 工作
        MOVAM   TM3_CTL1        ;; RFC IRQ 来自 TM0
        CLR     TM3_CTL2
        MOVLA   B'01000100'     ;; RFC，PH0X2，T/2:T/2
        MOVAM   TM0_CTL
        MOVLA   B'00111111'
        MOVAM   TM0_LA
        BC      TM0_CTL,6

        MOVLA   B'00000001'     ;; 设置 TM0 IRQ 掩膜
        MOVAM   IRQM
```

```
        BS      IRQM_CTL,7
        BS      TM3_CTL1,7
        BS      TM0_CTL1,7
                CLR             IRQF
        LGOTO   $
```

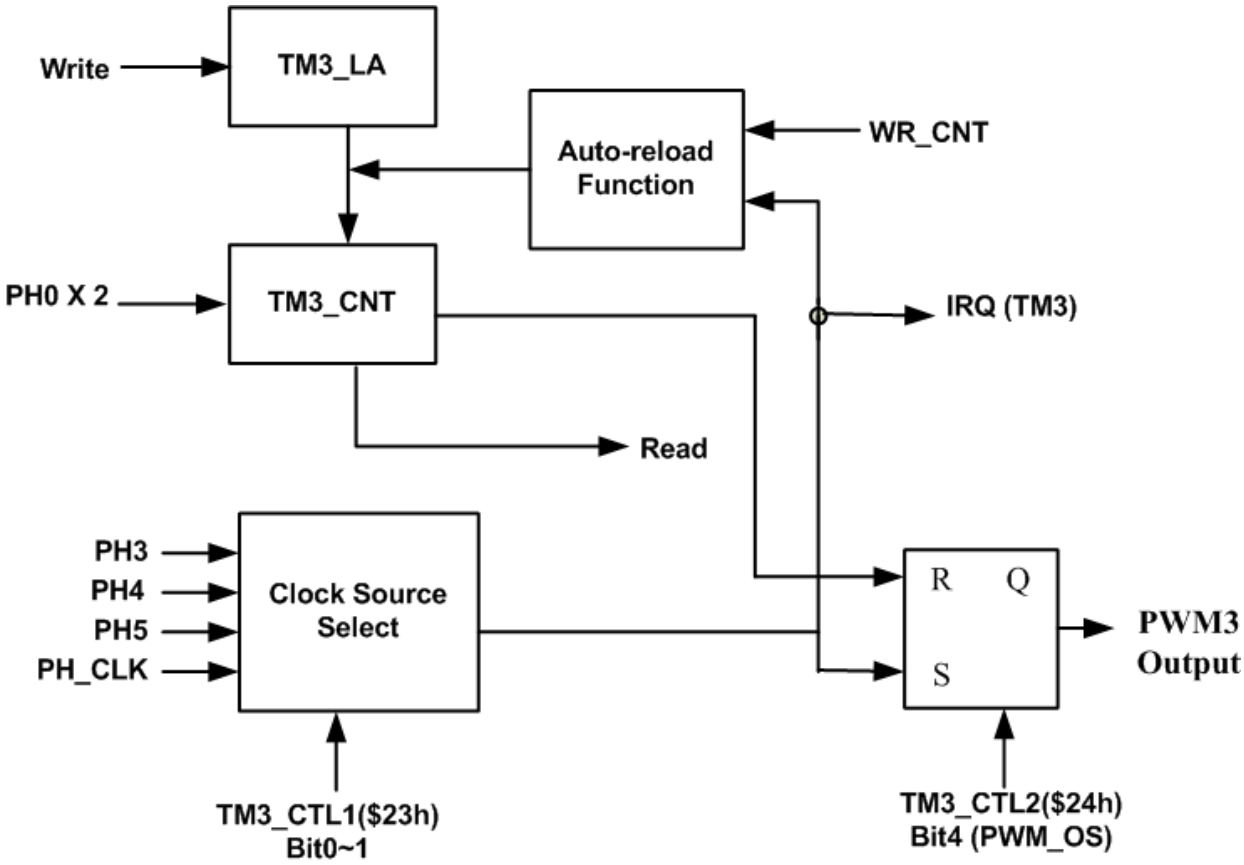


图.3.4.28 TM3 作为 PWM 的示意图

图.3.4.28 PWM2（TM3 PWM）输出示例

```
        INC     'MK9A50P.inc'   ;; TM3 PWM 模式
                                ;; Period=PH7，占空比来自 TM3
#DEFINE RAM_80  80H
        ORG     00              ;;
        LGOTO   START

INT:    ORG     004
        MOVLA   0x7B            ;; 清除 TM3
        MOVAM   IRQF
        INC     PD_DAT,m        ;; 检测 TM3 IRQ
        MOVLA   0x01
        ADD     TM3_LA          ;; 改变 TM3 PWM 占空比
        NOP
        IRETI

        ORG     100h
```

```
START:  CLR     STATUS
        MOVLA   02h
        MOVAM   LBASDT          ;; Com5~7 作为 I/O 口工作
        BC      SYS_CTL,b1      ;;  快时钟开启
        NOP
        BS      SYS_CTL,7       ;; CPU 时钟  = FCLK
        NOP
        CLR     RAM_80
        CLR     PA_DIR          ;; PA 输出
        CLR     PAD_CTL1        ;; PD 作为 I/O 口工作
        CLR     PD_DIR          ;; PD 输出
        ;;MOVLA  B’00110000”     ;; PA3 = PWM2 输出
        ;;MOVAM  PAD_CTL2
        MOVLA   B’01011000’     ;; PWM 模式，PWM 占空比=ph3
        MOVAM   TM3_CTL1
        MOVLA   B’00000001’
        MOVAM   TM3_CTL2
        MOVLA   0xE0            ;; PWM 占空比计数器
        MOVAM   TM3_LA
        BC      TM3_CTL1,6
        NOP
        MOVLA   B’00000100’     ;;  设置 TM3 IRQ 掩膜
        MOVAM   IRQM
        CLR     IRQF
        BS      IRQM_CTL,7
        BS      TM3_CTL1,7
        LGOTO   $
```

3.5. 看门狗定时器（WTD）

WDT是预防软件故障的定时器，或预防软件跳到一个不可预知结果的未知位置。WDT的源时钟是低速时钟。用户首先应通过配置寄存器的bit 5（WDTE）使能或禁止看门狗定时器。

WDT CTL ($3Bh):

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

- Bit7（WDTEN）：看门狗定时器使能/禁止位
  - 0：WDT禁止
  - 1：WDT使能

<注> Bit6~5（TEST1/TEST0）为测试保留位

- Bit2~0（PRE2~0）WDT预分配位

Bit2	Bit1	Bit0	WDT预分配率
PRE2	PRE1	PRE0	
0	0	0	Twdt
0	0	1	Twdt X 2
0	1	0	Twdt X 4
0	1	1	Twdt X 8
1	0	0	Twdt X 16
1	0	1	Twdt X 32
1	1	0	Twdt X 64
1	1	1	Twdt X 128（2’S 按键复位模式）

CONFIG		OSC 类型	Twdt
SOSC1	SOSC0		
0	0	LP（低速）	Twdt = Tsystem clock X 512
0	1	NO	Twdt = 15.6 mS
1	0	外部 RC	Twdt = Tsystem clock X 512
1	1	内部 RC	Twdt = 15.6 mS

表 3.5.1 慢时钟类型与WDT之间的关系

4. I/O 口及其他控制功能

4.1. I/O 口

有 4 个 I/O 口组 PA，PC，PD 及 PD 用于输入或输出数据，每一个 I/O 口有不同的定义，它们大多与其他功能分享脚位。A 口（PA）是双向 I/O 口，通过设置有上拉，下拉，开漏及脚位改变唤醒功能。C 口（PC）是双向 I/O 口，通过设置有下拉，开漏及脚位改变唤醒功能。D 口（PD）及 E 口（PE）是 I/O 口带下拉，开漏。

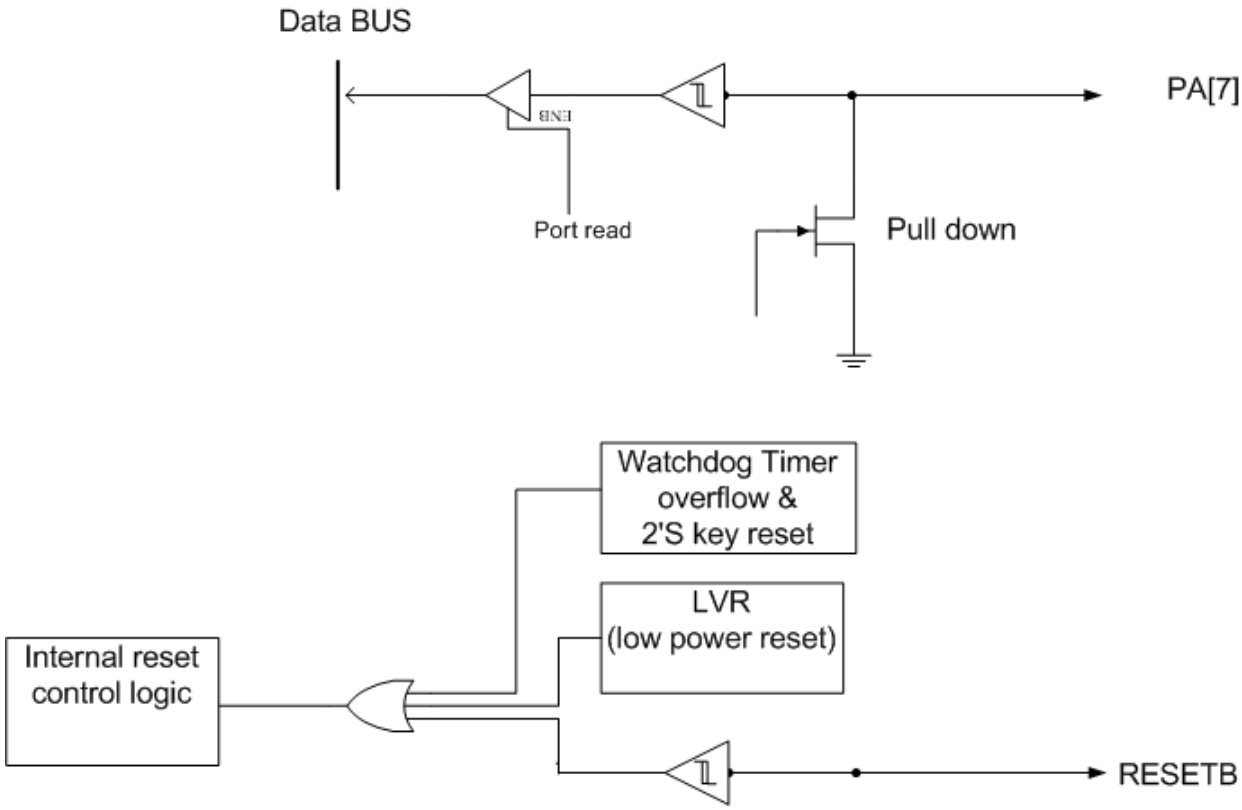


图.4.1.1-1　RESETB（PA7）结构

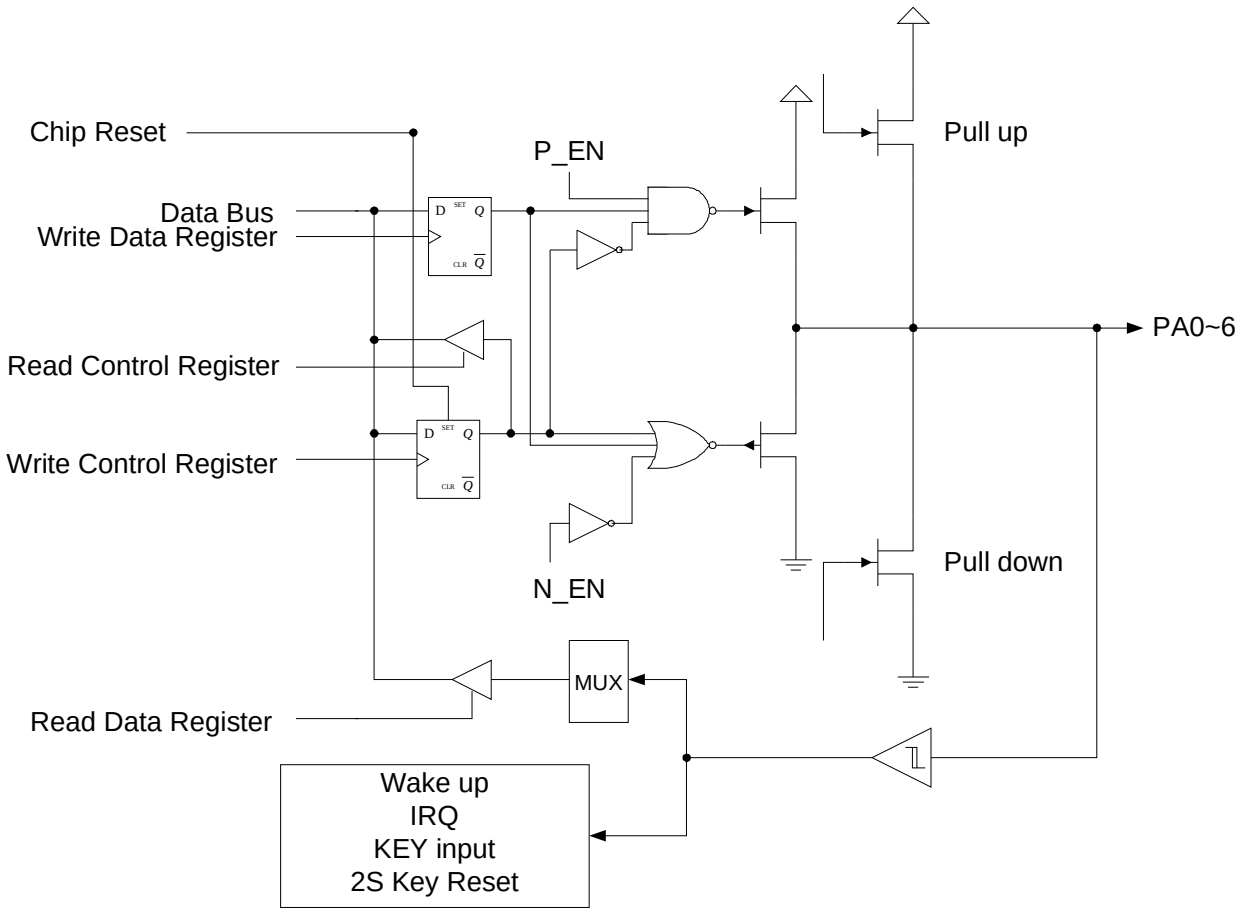


图.4.1.1-2 PORT A 结构

4.1.1. Port A（PA）

A 口是该芯片的唯一双向 I/O 口。它有上拉，下拉，开漏及脚位唤醒功能。如果用户设置配置位，它也可以替代RESET脚位去执行通电复位。有6个寄存器去设置8个I/O口，分别是PA_DIR，PA_CTL，PA_DAT，WAKE_UP，PA_PUD1 及 PA_PUD2。寄存器定义如下：

PA_DIR ($05h):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_DIR	--	CA6	CA5	CA4	CA3	CA2	CA1	CA0

- Bit6~0（CA6~0）：设置PA作为输入口或输出口（PA[7]只可作为输入口）
  0：输出口
  1：输入口

PA_CTL ($06h):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_CTL	--	KI6/IN6	KI5/IN5	KI4/IN4	KI3/IN3	KI2/IN2	KI1/IN1	KI0/IN0

- Bit7~0（CA6~0）：选择PA输入作为KI模式或I/O输入口使用

  0：PA作为I/O输入模式，使用脚位沿IRQ（正常，暂停模式及2'S按键复位功能）

  1：PA作为KI输入模式，使用KEY唤醒IRQ（正常或暂停模式）

PA_CTL	PA_DIR	WAKE_UP	FUNCTION
KIn/INn	CAn	ENn	N=0~6
1	X	X	按键输入模式
0	1	1	输入脚位沿唤醒模式

PA_EDGE ($2Dh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_EDGE	EDGE7	EDGE6	EDGE5	EDG4	EDG3	EDGE2	EDGE1	EDGE0

- Bit7~0（EDGE~0）：脚位PA7~0上升/下降唤醒控制位

  1：下降沿唤醒

  0：上升沿唤醒

PA_WAKE_UP ($07h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_WAKE_UP	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0

- Bit7~0（PA7~0）：PA7~0脚位改变唤醒控制位

  0：唤醒禁止

  1：唤醒使能（仅PA在PAD_CTL1 ($13h)被设置为输入时活动）

CAn (PA_DIR)	ENn (WAKE_UP)	KI（脚位沿唤醒功能）
1	1	ON
X	0	OFF
0	1	OFF

PA_PUD1 ($08h) & PA_PUD2 ($09h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_PUD1	A3-2	A3-1	A2-2	A2-1	A1-2	A1-1	A0-2	A0-1
PA_PUD2	--	A7-1	A6-2	A6-1	A5-2	A5-1	A4-2	A4-1

- 当设置PA作为I/O口（PAD_CTL2）及方向是输入（PA_DIR）时，则

  An-1=1 -> Pan下拉

  An-2=1 -> Pan上拉

- 当设置PA作为I/O口（PAD_CTL2）及方向是输出（PA_DIR）时，则

  An-1=1 -> PMOS开漏开

  An-2=1 -> NMOS开漏开

这两个寄存器用于设置PA有上拉或下拉寄存器。但这仅在PAD_CTL2被设置为I/O口及PA_DIR被设置为输入时可用。他们之间的关系如下表。

KIn/PAn (PAD_CTL2)	CAn (PA_DIR)	An-2	An-1	上拉	下拉	PMOS 开漏	NMOS 开漏	描述 Pan（n=0~6）
I/O#	1	0	0	OFF	OFF	OFF	OFF	PAn 是输入口
I/O#	1	0	1	OFF	**ON**	OFF	OFF	PAn 是输入口
I/O#	1	1	X	**ON**	OFF	OFF	OFF	PAn 是输入口
I/O#	0	0	0	OFF	OFF	ON	ON	PAn 是正常输出
I/O#	0	0	1	OFF	OFF	ON	OFF	PAn 是 nmos 开漏输出口
I/O#	0	1	X	OFF	OFF	OFF	ON	PAn 是 pmos 开漏输出口
I , KI	1	0	0	OFF	ON/OFF	OFF	OFF	PAn 是 KI 输入模式

<注> (1) PA[7]与RESETB分享，RESETB只可作为输入口使用。因此，只可设置下拉功能，如下表。

(2) 其他#包括（ELP，ELC，PWM，REM，BZ，BZM）)

(3) I/O#包括（PA I/O模式，CAPT1A & CAPT1B）

PA7/KI7	A7-2	A7-1	下拉	PMOS 开漏	NMOS 开漏	PA7
PA7	0	0	OFF	OFF	OFF	PA7 是输入口
PA7	0	1	**ON**	OFF	OFF	PA7 是输入口
PA7	1	X	OFF	OFF	OFF	PA7 是输入口
KI7	0	0	**ON/OFF**	OFF	OFF	PA7 是 KI 输入

PA.. _DAT($0Ah):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PA_DAT	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0

- Bit6~0（PA6~0）：PA输入或输出数据位
- Bit7：仅输入数据

<Note> PA[5]~PA[7]与OSCOUT，OSCIN及RESETB分享脚位。如果用户要把这些脚位作为PA使用，则配置bit 3~2（FOSC1及FOSC0）必须设置为（0，1），bit 8（RST_DEF）必须设置为0作为正常输入口。

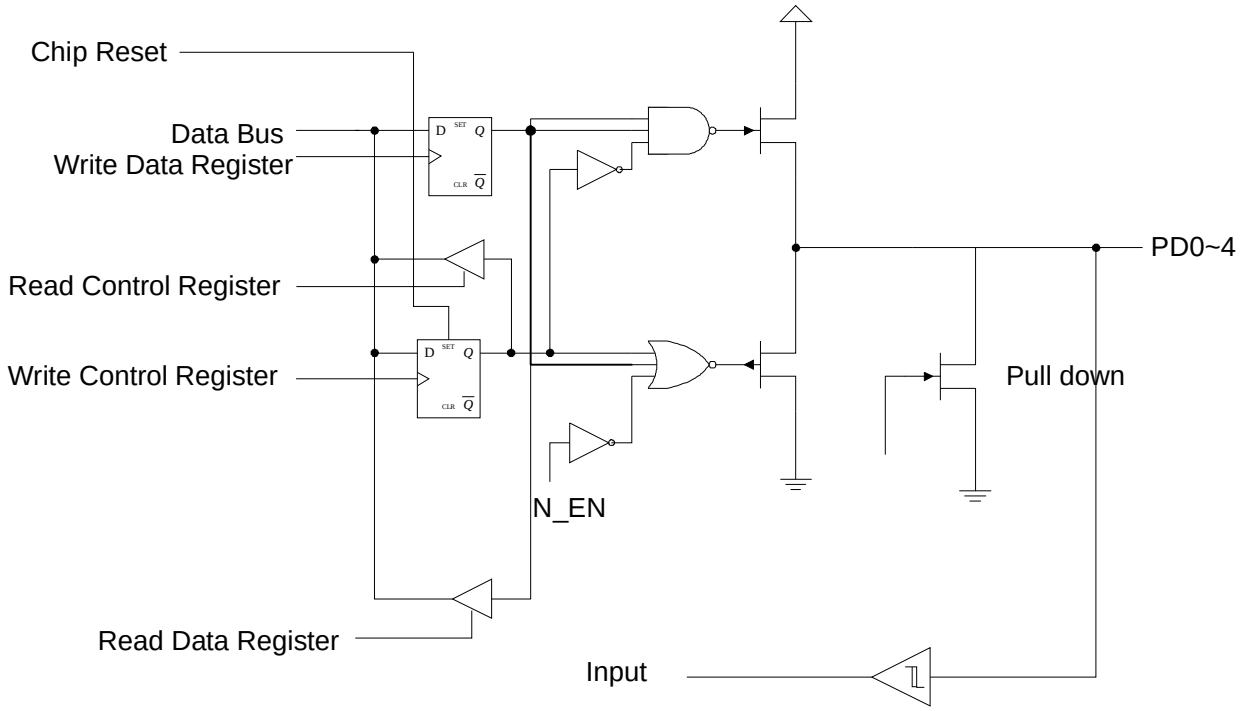


图.4.1.2 PORT C 结构

4.1.2 Port C（PC）

有2个寄存器设置C口的属性，分别是PC_CTL及PC_DAT。C口作为I/O口使用时只是输入口。

PC_EDGE ($1Eh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_EDGE	EDGE7	EDGE6	EDGE5	EDG4	EDG3	EDGE2	EDGE1	EDGE0

- Bit7~0（EDGE~0）：脚位PC7~0上升/下降唤醒控制位

  1：下降沿唤醒

  0：上升沿唤醒

PC_WAKE_UP ($1Dh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_WAKE_UP	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0

- Bit7~0（PC7~0）：脚位PC7~0脚位改变唤醒控制位

  0：唤醒禁止

  1：唤醒使能（仅PA在PAD_CTL1 ($14h)被设置为输入时活动）

CAn (PC_DIR)	ENn (WAKE_UP)	KI（脚位沿唤醒功能）
1	1	ON
X	0	OFF
0	1	OFF

PC_CTL ($0Bh): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_CTL	KI7/IO7	KI6/IO6	KI5/IO5	KI4/IO4	KI3/IO3	KI2/IO2	KI1/IO1	KI0/IO0

- Bit1~0（KI1/IO1）：PC1~0作为KI或I/O输入模式工作（仅输入）
  0：I/O输入模式
  1：KEY输入
- Bit1~0（KIn/Ion）：PC7~0作为KI或I/O输入模式工作（仅输入）
  0：PA作为I/O输入模式，使用脚位沿IRQ（正常，暂停模式及2'S按键复位功能）
  1：PA作为KI输入模式，使用KEY唤醒IRQ（正常或暂停模式）

PC_CTL	PC_DIR	WAKE_UP	功能
KIn/INn	CAn	ENn	
1	X	X	按键输入模式
0	1	1	输入脚位沿唤醒模式

PS. PC[0~2] & PC[4]不使用KEY STROBE功能。

PC_DIR ($0Ch): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_DIR	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0

- Bit4~0（DC4~0）：设置PC作为输入口或输出口
  0：输出口
  1：输入口

PC_PUD ($0Dh) : (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_PUD	CD7	CD6	CD5	CD4	CD3	CD2	CD1	CD0

<注> 当这些分享脚位被设置为PC，有PMOS开漏或NMOS开漏可供选择。设置方法如下表：

PC_DIR DCn	PC_PUD CDn	下拉	PMOS 开漏	描述 PCn (n=0~7)
1	0	OFF	OFF	PCn 是输入口
1	1	**ON**	OFF	PCn 是输入口
0	0	ON	ON	PCn 是正常输出
0	1	OFF	ON	PCn 是 PMOS 开漏输出口

PC_DAT($0Eh): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PC_DAT	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

- Bit7~0（PC7~0）：PC输入数据位

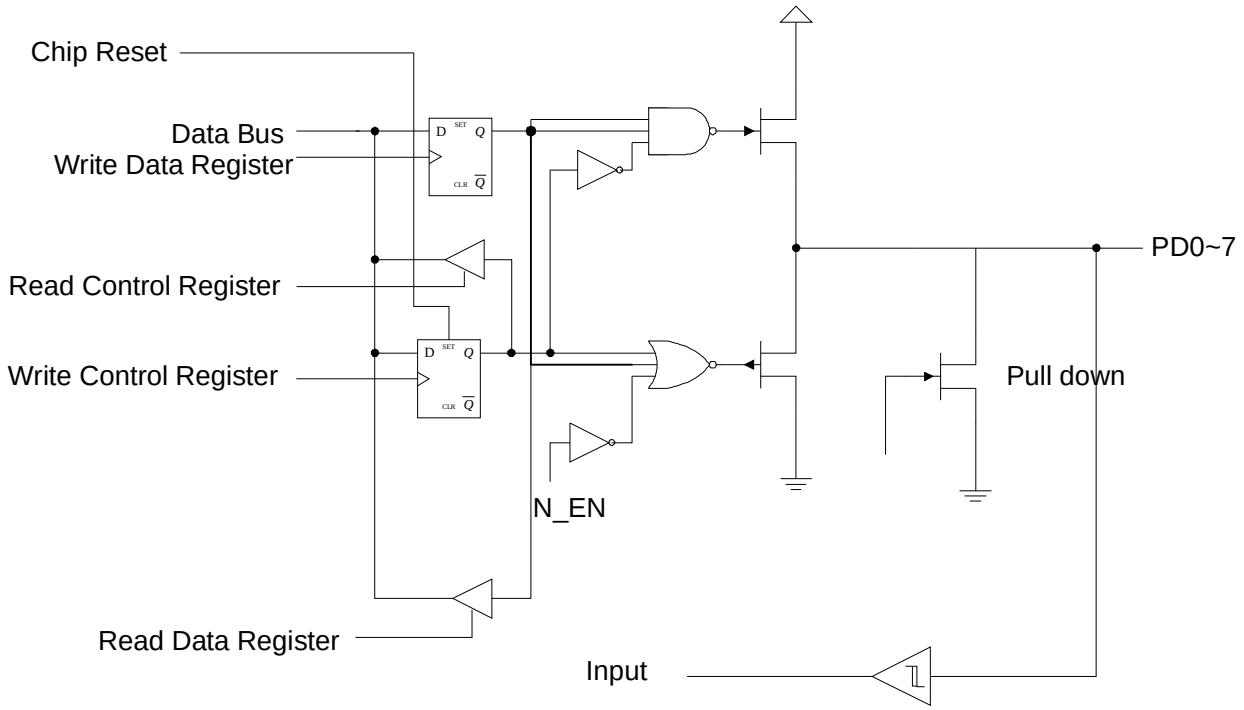


图.4.1.3 PORT D 结构

4.1.3 Port D（PD）

有3个寄存器可设置D口的属性，分别是PD_DIR，PD_PUD及PD_DAT。D口作为I/O口使用时只是输出口。

PD_DIR ($0Fh): (R/W) (default =11111111b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PD_DIR	DD7	DD6	DD5	DD4	DD3	DD2	DD1	DD0

- Bit7~0（DD7~0）：设置PD作为输入口或输出口
  - 0：输出口
  - 1：输入口

PD_PUD ($10h) : (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PD_PUD	CD7	CD6	CD5	CD4	CD3	CD2	CD1	CD0

<注>当这些分享脚位被设置为PD，有PMOS开漏或NMOS开漏可供选择。设置方法如下表：

DDn (PD_DIR)	CDn	下拉	PMOS 开漏	描述 PDn (n=0,1,2,3,5)
1	0	OFF	OFF	PDn 是输入口
1	1	**ON**	OFF	PDn 是输入口
0	0	ON	ON	PDn 是正常输出
0	1	OFF	ON	PDn 是 PMOS 开漏输出口

DDn (PD_DIR)	CDn	下拉	PMOS 开漏	描述 PDn (n=4)
1	CD4=0	OFF	OFF	PDn 是输入口
1	CD3=4	**OFF**	OFF	PDn 是输入口
1	CD3=1	**ON**	OFF	PDn 是输入口
0	CD4=0	ON	ON	PDn 是正常输出
0	CD4=1	OFF	ON	PDn 是 PMOS 开漏输出口

PD_CTL ($11h): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PD_CTL	KI7/IO7	KI6/IO6	KI5/IO5	KI4/IO4	KI3/IO3	KI2/IO2	KI1/IO1	KI0/IO0

- Bit4~3（DC7~0）：PD作为KEY或I/O输入模式工作（仅输入）
  - 0：I/O输入口
  - 1：按键输入口

PD_DAT($12h):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PD_DAT	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0

- Bit7~0（PD7~0）：PD输出数据位

4.1.4 Port E（PE）

有3个寄存器可设置E口的属性，分别是PE_DIR，PE_PUD及PE_DAT。E口作为I/O口使用时只是输出口。

PE_DIR ($1Ah): (R/W) (default =11111111b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PE_DIR	DE7	DE6	DE5	DE4	DE3	DE2	DE1	DE0

- Bit7~0（DE7~0）：设置PD作为输入口或输出口
  0：输出口
  1：输入口

PE_PUD ($1Bh) : (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PE_PUD	CE7	CE6	CE5	CE4	CE3	CE2	CE1	CE0

<注>当这些分享脚位被设置为PE，有PMOS开漏或NMOS开漏可供选择。设置方法如下表：

DEn (PE_DIR)	Dn-1	下拉	PMOS 开漏	描述 PEDn (n=0~7)
1	0	OFF	OFF	PEn 是输入口
1	1	**ON**	OFF	PEn 是输入口
0	0	ON	ON	PEn 是正常输出
0	1	OFF	ON	PEn 是 pmos 开漏输出口

PE.._DAT($1Ch): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PE_DAT	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0

- Bit7~0（PE7~0）：PE输出数据位

4.2. 定义分享脚位

PAD_CTL1 ($13h): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD_CTL1	SEG40/ PD[7]	SEG39/ PD[6]	SEG38/ PD[5]	SEG37/ PD[4]	SEG36/ PD[3]	SEG35/ PD[2]	SEG34/ PD[1]	SG33/ PD[0]

<注> 此寄存器用于设置分享脚位功能，设置如下表。

Bit	Bitn=1	Bitn=0
0	SEG33	PD[0]
1	SEG34	PD[1]
2	SEG35	PD[2]
3	SEG36	PD[3] / PWM3
4	SEG37	PD[4] + CAPT2A
5	SEG38	PD[5] + INT
6	SEG39	PD[6] / ELP
7	SEG40	PD[7] / ELC

Bit \value	11	10	01	00
(PH_CTL.EL_P),B6	Don't	ELP	SEG39	PD6
(PH_CTL.EL_P),B7	Don't	ELC	SEG40	PD7

PAD_CTL2 ($14h): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD_CTL2		C6	C5	C4	C3	C2	C1	C0

<注> 此寄存器用于设置分享脚位功能，设置如下表。

Bit \value	0	1
C0	PA0	CAP
C1	PA1	REF
C2	PA2	SEN0
C3	PA4	BZ
C6	PA5	BZM

Bit \value	11	10	01	00
C5-C4	PWM2	SEN1	REM	PA3+CAPT1A

4.2-1 PD[5] 唤醒（IRQ）

```
#INCLUDE   "MK9A50P.INC"          ;; PD[5]脚位沿唤醒
                                  ;; 睡眠模式或暂停模式
           ORG       0x00
           LGOTO     INITIAL
INT:       ORG       0x04
```

PAD_CTL3 ($15h): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD_CTL3	EDGE	--			--	SEN1_ON	SEN0_ON	REF_ON

<注> 此寄存器用于设置分享脚位功能，设置如下表。

- Bit7（EDGE）：PD[5]脚位中断控制
  - 0：上升沿
  - 1：下降沿
- Bit2（SEN1_ON）：设置SEN1脚位输出控制（仅RFC模式）
  - 0：输入禁止
  - 1：输入使能
- Bit1（SEN0_ON）：设置SEN0脚位输出控制（仅RFC模式）
  - 0：输入禁止
  - 1：输入使能
- Bit0（REF_ON）：设置REF脚位输出控制（仅RFC模式）
  - 0：输入禁止
  - 1：输入使能

PAD_CTL4 ($16h): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD_CTL4	SEG32/ PE[7]	SEG31/ PE[6]	SEG30/ PE[5]	SEG29/ PE[4]	SEG28/ PE[3]	SEG27/ PE[2]	SEG26/ PE[1]	SG25/ PE[0]

<注> 此寄存器用于设置分享脚位功能，设置如下表。

Bit	Bitn=1	Bitn=0
0	SEG25	PE[0]
1	SEG26	PE[1]
2	SEG27	PE[2]
3	SEG28	PE[3]
4	SEG29	PE[4]
5	SEG30	PE[5]
6	SEG31	PE[6]
7	SEG32	PE[7]

PAD_CTL5 ($28h): (R/W) (default =00000000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD_CTL5	--	--	--	SEG23/ PC[6]	SEG24/ PC[7]	SEG42/ PC[3]	SEG41/ PC[4]	PWM3/ PD[3]

<注> 此寄存器用于设置分享脚位功能，设置如下表。

Bit	Bitn=1	Bitn=0
0	PWM3	PD[3]
1	SEG41	PC[4]
2	SEG42	PC[3]
3	SEG24	PC[7]
4	SEG23	PC[6]

4.3. 按键选通功能

按键扫描功能使用部份分段帧频率周期去输出扫描时间。此功能被限制在正常模式或暂停模式。

寄存器位定义如下：

STROBE($34h) ：选通控制 (R/W) (default =000x0000b)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STROBE	KIEN1	KIEN0	KOAEN	KOEN	KO3	KO2	KO1	KO0

- Bit7~6（KIEN1~0）：按键模式选择

Bit \value	11	10	01	00
KIEN1~0	硬件模式 2	硬件模式 1	软件模式	OFF

模式/功能	SEG1~16	PORT PA0~6 & PC0~7	PORT PD3~4	IRQ
硬件模式 2	X	下拉使能	X	V
硬件模式 1	Hi 输出	下拉使能	X	V
软件模式	X	下拉使能	下拉使能	X
KOAEN	SEG1~16：Hi 输出	X	X	X
KOEN	SEGn：Hi 输出 其他：浮动	X	X	X

- Bit5（KOAEN）：分段输出控制
  - 0：SEG1~16：LCD 输出
  - 1：SEG1~16：Hi输出
- Bit4（KOEN）：分段输出控制
  - 0：SEG1~16：LCD 输出
  - 1：SEGn：Hi输出，其他的SEG：浮动输出

KIn/ PAn	CAn	An-2	An-1	STBEN	HSCNE	上拉	下拉	PMOS 开漏	NMOS 开漏	描述 PAn (n=0~6)
I , KI	1	0	0	1	X	OFF	ON	OFF	OFF	PAn 是 KI 下拉输入
I , KI	1	0	0	0	1	OFF	ON	OFF	OFF	PAn 是 KI 下拉输入
I , KI	1	0	0	0	0	OFF	OFF	OFF	OFF	PAn 是浮动

PA_CTL：PA KI / INPUT（I/O）控制寄存器

PA_dir：PA IN / OUT 控制寄存器

- Bit3~0（KO3~0）：选择哪一个选通脚位去输出信号见下表（f – 浮动）

Bit3~0	KS1	KS2	KS3	KS4	KS5	KS6	KS7	KS8	KS9	KS10	KS11	KS12	KS13	KS14	KS15	KS16
0000	HI	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f
0001	f	HI	f	f	f	f	f	f	f	f	f	f	f	f	f	f
0010	f	f	HI	f	f	f	f	f	f	f	f	f	f	f	f	f
0011	f	f	f	HI	f	f	f	f	f	f	f	f	f	f	f	f
0100	f	f	f	f	HI	f	f	f	f	f	f	f	f	f	f	f
0101	f	f	f	f	f	HI	f	f	f	f	f	f	f	f	f	f
0110	f	f	f	f	f	f	HI	f	f	f	f	f	f	f	f	f
0111	f	f	f	f	f	f	f	HI	f	f	f	f	f	f	f	f
1000	f	f	f	f	f	f	f	f	HI	f	f	f	f	f	f	f
1001	f	f	f	f	f	f	f	f	f	HI	f	f	f	f	f	f
1010	f	f	f	f	f	f	f	f	f	f	HI	f	f	f	f	f
1011	f	f	f	f	f	f	f	f	f	f	f	HI	f	f	f	f
1100	f	f	f	f	f	f	f	f	f	f	f	f	HI	f	f	f
1101	f	f	f	f	f	f	f	f	f	f	f	f	f	HI	f	f
1110	f	f	f	f	f	f	f	f	f	f	f	f	f	f	HI	f
1111	f	f	F	f	f	f	f	f	f	f	f	f	f	f	f	HI

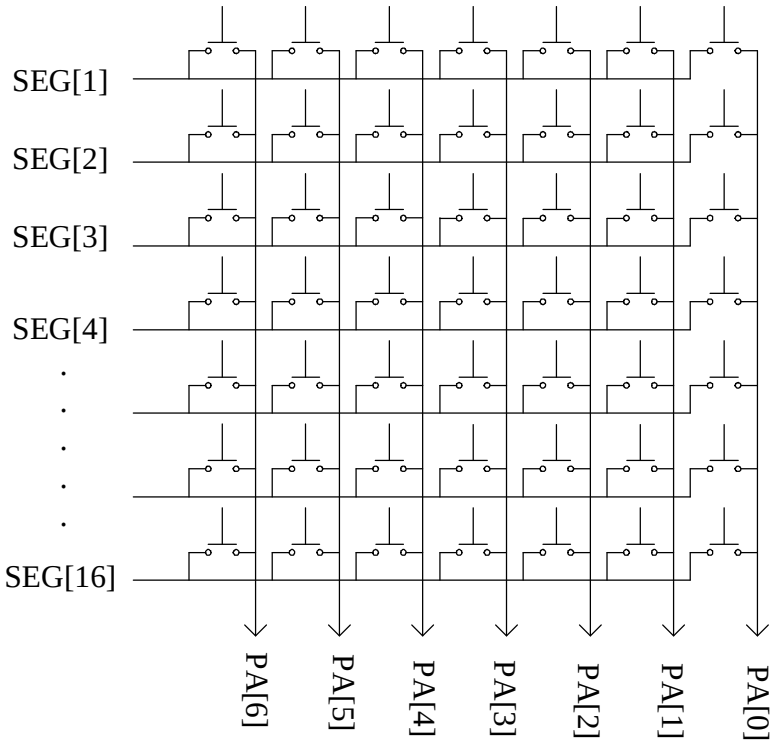


图.4.3.1　软件模式（PA[0-6] 输入，SEG-输出）

4.3.1-1 硬件模式 1 模式示例（PA[6:0]-输入，SEG[16:1]-输出）

```
#INCLUDE   "MK9A50P.INC"                ;; 硬件按键扫描低功率消耗
                                        ;; 暂停模式，PA IRQ唤醒
                                        ;; 键矩阵：M X N（PA X SEG）
                                        ;; PA[6:0]输入带下拉
                                        ;; SEG[16~1]扫描输出带P-开漏
#DEFINE    KEY_DATA     80h
#DEFINE    AVAIL_KEY    81h             ;; 可用按键（PA6~0）= M
#DEFINE    KEY_NUM      82h             ;; KEY扫描环路号码 = N
#DEFINE    ACC_REG      83h
#DEFINE    STATUS_REG   84h
           ORG          0x00
           LGOTO        INITIAL
ORG        0x04
           MOVAM        ACC_REG         ;; 按ACC
           MOV          STATUS
           MOVAM        STATUS_REG      ;; 按STATUS
           MOVLA        B'10111111'
           MOVAM        IRQF            ;; 清除PA irq
           MOVLA        B'01010000'     ;; 软件读取按键 & 硬件模式1模式
           MOVAM        STROBE          ;; PA[6~0]下拉 & 输入缓冲器打开
KEY_SCAN   ;; NOP                       ;; 等待按键加载
           MOV          PA_DAT
           MOVAM        KEY_DAT,m
           MOV          AVAIL_KEY
           AND          KEY_DAT,m       ;; 可用按键数据
           TMSC         KEY_DAT
           LGOTO        KEY_END
           MOV          KEY_NUM         ;; 按键扫描号码
           TMCOMPE      STROBE          ;; 比较STROBE=00011111
           LGOTO        KEY_END
           INC          STROBE,m
           LGOTO        KEY_SCAN
KRY_END    …….
           MOVLA        B'10000000'     ;; 使能硬件按键扫描
           MOVAM        STROBE          ;; PA[6~0]下拉 & 输入缓冲器开启关闭
                                        ;; 按KEY有无功率消耗
           MOV          STATUS_REG
           MOVAM        STATUS          ;; 取出STATUS
```

```
            MOV         ACC_REG     ;; 取出ACC
            RETI

            ORG         100h        ;; 主程序，按KEY有无功率消耗
INITIAL     CLR         STATUS
            MOVLA       B'00011111' ;; KEY号码 =16（SEG16~1）
            MOVAM       KEY_NUM
            MOVLA       0xFF
            MOVAM       PA_DIR      ;; PA输入
            MOVLA       0x7F        ;; PA6~0作为KEY输入工作
            MOVAM       PA_CTL
            MOVLA       B'10000000' ;; 硬件模式1模式 – 自动按键扫描
            MOVAM       STROBE
            CLR         IRQF
            MOVLA       B'01000000' ;; 使能PORT A  irq
            MOVAM       IRQM
            BS          IRQM_CTL,7  ;; 使能IRQ
            NOP
LOOP        ……
            BS          SYS_CTL,6   ;; 暂停模式
            ……
            ……
            LGOTO       LOOP
```

4.3.1-2 硬件模式 1 模式示例（PA[6:0]-输入，SEG[16:1]-输出）

```
#INCLUDE    "MK9A50P.INC"           ;; 软件按键扫描
                                    ;; 正常模式，轮询
                                    ;; 键矩阵：M X N（PA X SEG）
                                    ;; PA[6:0]输入带下拉
                                    ;; SEG[16~1]按键输出带P-开漏
#DEFINE     KEY_DATA    80h
#DEFINE     AVAIL_KEY   81h         ;; 可用按键（PA6~0）= M
#DEFINE     KEY_NUM     82h         ;; KEY扫描环路号码 = N
#DEFINE     ACC_REG     83h
#DEFINE     STATUS_REG  84h
            ORG         0x00
            LGOTO       INITIAL
ORG         0x04
```

```
            ……
            IRET

KEY         MOVLA       B’01010000’   ;; 软件读取按键 & 硬件模式1模式
            MOVAM       STROBE        ;; PA[6~0]下拉 & 输入缓冲器开启
KEY_SCAN    ;; NOP                    ;; 等待按键加载
            MOV         PA_DAT
            MOVAM       KEY_DAT,m
            MOV         AVAIL_KEY
            AND         KEY_DAT,m     ;; 可用按键数据
            TMSC        KEY_DAT
            LGOTO       KEY_END
            MOV         KEY_NUM       ;; 按键扫描号码
            TMCOMPE     STROBE        ;; 比较STROBE=00011111
            LGOTO       KEY_END
            INC         STROBE,m
            LGOTO       KEY_SCAN
KRY_END     …….
            MOVLA       B’00000000’   ;; 按键扫描结束
            MOVAM       STROBE        ;; PA[6~0]下拉 & 输入缓冲器关闭
                                      ;; 按KEY有无功率消耗
            RET

            ORG         100h          ;; 主程序，按KEY有无功率消耗
INITIAL     CLR         STATUS
            MOVLA       B’00011111’   ;; KEY号码 =16（SEG16~1）
            MOVAM       KEY_NUM
            MOVLA       0xFF
            MOVAM       PA_DIR        ;; PA输入
            MOVLA       0x7F          ;; PA6~0作为KEY输入工作
            MOVAM       PA_CTL
            MOVLA       B’10000000’   ;; 硬件模式1模式 – 自动按键扫描
            MOVAM       STROBE
            NOP
LOOP        ……
            ……
            MOVLA       B’00100000’   ;; 软件模式模式 – 按键扫描全部，SEG=FFFFh
            MOVAM       STROBE
            MOV         PA_DAT
```

```
MOVAM       KEY_DAT
MOV         AVAIL_KEY
AND         KEY_DAT,m    ;; 按键按或NO?
TMSC        KEY_DAT      ;; 按KEY è 呼叫键子程序
LCALL       KEY
……
LGOTO       LOOP
```

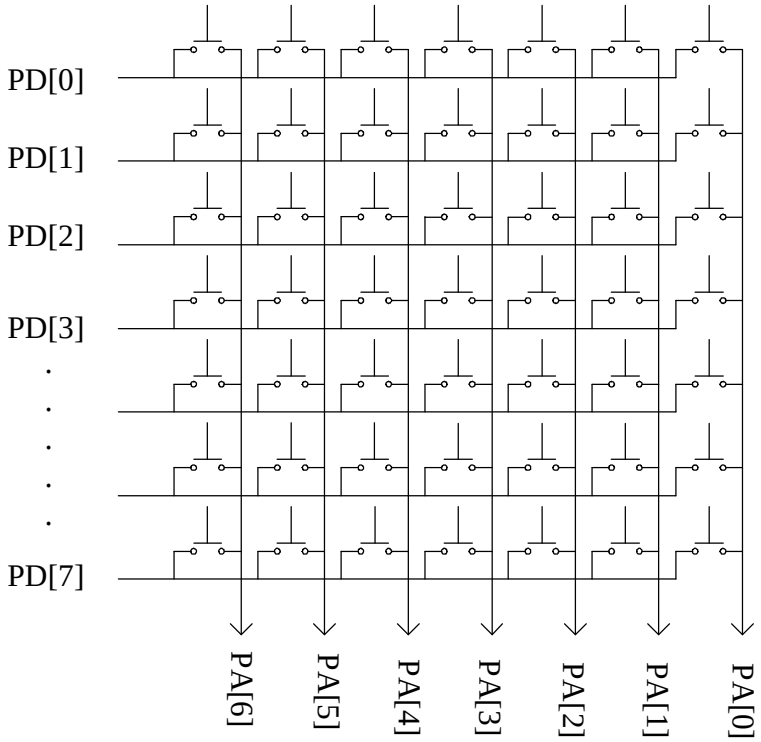


图.4.3.2 硬件模式 2，（PA-输入，PD-输出）

4.3.2-1 硬件模式 2 模式示例（PA[6:0]-输入，PD[7:0]-输出）

```
#INCLUDE   "MK9A50P.INC"          ;; 低功率消耗
                                  ;; 暂停 & PA IRQ唤醒（KI模式）
                                  ;; 键矩阵：M X N（PA X PD（或PC））
                                  ;; PA[6:0]输入带下拉
                                  ;; PC或PD pmos开漏输出

#DEFINE    KEY_DATA    80h
#DEFINE    AVAIL_KEY   81h        ;; 可用按键（PA6~0）
#DEFINE    ACC_REG     83h
#DEFINE    STATUS_REG  84h
           ORG         0x00
```

```
            LGOTO       INITIAL

ORG         004
            MOVAM       ACC_REG     ;; 按ACC
            MOV         STATUS
            MOVAM       STATUS_REG  ;; 按STATUS
            MOVLA       B'10111111'
            MOVAM       IRQF        ;; 清除PA irq
            MOVLA       B'00000001' ;; PD[0] – hi输出，PD[7:1] – 浮动输出
            MOVAM       PD_DAT
KEY_SCAN    ;; NOP                  ;; 等待按键加载
            MOV         PA_DAT
            MOVAM       KEY_DAT,m
            MOV         AVAIL_KEY
            AND         KEY_DAT,m   ;; 可用按键数据PA[6~0]
            TMSC        KEY_DAT
            LGOTO       KEY_END
            BTSC        PD_DAT, 7   ;; 当PD[7]=1，按键扫描将会停止
            LGOTO       KEY_END
            RL          PD_DAT,m    ;; 改变PD数据PD[7]ßPD[6]ß …….PD[1]ßPD[0]
            LGOTO       KEY_SCAN
KRY_END     …….
            MOV         STATUS_REG
            MOVAM       STATUS      ;; 取出STATUS
            MOV         ACC_REG     ;; 取出ACC
            IRET

            ORG         200h
INITIAL     CLR         STATUS
            MOVLA       0xFF
            MOVAM       PA_DIR      ;; PA输入
            MOVLA       0xFF
            MOVAM       PD_PUD      ;; 下拉输入或pmos开漏输出
            MOVLA       0x00
            MOVAM       PD_DIR      ;; PD pmos开漏输出
            MOVLA       0xFF        ;; PD=FFh
            MOVAM       PD_DAT      ;; 等待KEY按压
            MOVLA       0x7F        ;; PA6~0作为KI输入工作
            MOVAM       PA_CTL
```

```
            CLR         IRQF
            MOVLA       B'01000000'   ;; 使能PORT A  irq
            MOVAM       IRQM
            BS          IRQM_CTL,7    ;; 使能IRQ
            NOP
LOOP        ……
            ……
            BS          SYS_CTL,6     ;; 暂停模式
            ……
            LGOTO       LOOP
```

4.3.2-2 硬件模式 2 模式示例（PA[6:0]-输入，PD[7:0]-输出）

```
#INCLUDE    "MK9A50P.INC"             ;; 功率消耗更大，仅正常
                                      ;; 正常模式 & 轮询KEY（KI模式）
                                      ;; PC或PD pmos开漏输出
                                      ;; PA[6:0]输入带下拉
#DEFINE     KEY_DATA    80h
#DEFINE     AVAIL_KEY   81h           ;; 可用按键（PA6~0）
            ORG         0x00
            LGOTO       INITIAL
ORG         004
            …….
            IRET
ORG         100
KEY         MOVLA       B'00000001'   ;; PD[0] – hi输出，PD[7:1] – 浮动输出
            MOVAM       PD_DAT
KEY_SCAN    ;; NOP                    ;; 等待按键加载
            MOV         PA_DAT
            MOVAM       KEY_DAT,m
            MOV         AVAIL_KEY
            AND         KEY_DAT,m     ;; 可用按键数据PA[6~0]
            TMSC        KEY_DAT
            LGOTO       KEY_END
            BTSC        PD_DAT, 7     ;; 当PD[7]=1，按键扫描将会停止
            LGOTO       KEY_END
            RL          PD_DAT,m      ;; 改变PD数据PD[7] ß PD[6] ß …PD[1] ß PD[0] ß
                                      1'b1
            LGOTO       KEY_SCAN
```

```
KRY_END   …….
          RET

          ORG       200h
INITIAL   CLR       STATUS
          MOVLA     0xFF
          MOVAM     PA_DIR      ;; PA输入
          MOVLA     0xFF
          MOVAM     PD_PUD      ;; 下拉输入或pmos开漏输出
          MOVLA     0x00
          MOVAM     PD_DIR      ;; PD pmos开漏输出
          MOVAM     PD_DAT      ;; PD7~0浮动
                                ;; 低功率消耗
          MOVLA     0x7F        ;; PA6~0作为KEY输入工作
          MOVAM     PA_CTL
          MOVLA     B'11000000' ;; 硬件模式2模式 – 自动按键扫描
          MOVAM     STROBE
          NOP
LOOP      ……
          MOVLA     0xFF
          MOVAM     PD_DAT
          MOV       PA_DAT
          MOVAM     KEY_DAT
          MOV       AVAIL_KEY
          AND       KEY_DAT,m
          BTSC      KEY_DAT     ;; 按KEY è 呼叫键子程序
          LCALL     KEY
          ……
          LGOTO     LOOP
```

4.3.2-3 硬件模式 2 模式示例（PA[6:0]-输入，PD[7:0]-输出）

```
#INCLUDE  "MK9A50P.INC"       ;; 低功率消耗
                              ;; 睡眠模式 & PA脚位沿唤醒（I/O模式）
                              ;; PC或PD pmos开漏输出
                              ;; 暂停（睡眠）模式唤醒使用PA脚位沿
                              ;; 唤醒后，低功率消耗使用PA KI模式
#DEFINE   KEY_DATA  80h
#DEFINE   AVAIL_KEY 81h       ;; 可用按键（PA6~0）
```

```
        ORG     0x00
        LGOTO   INITIAL

ORG     004
        MOVAM   ACC_REG     ;; 按ACC
        MOV     STATUS
        MOVAM   STATUS_REG  ;; 按STATUS
        MOVLA   0x00
        MOVAM   WAKE_UP     ;; 脚位沿唤醒关闭
        MOVLA   0x7F        ;; PA6~0作为KI输入工作
        MOVAM   PA_CTL
        MOVLA   B'00000001' ;; PD[0] -- hi 输出，PD[7:1] – 浮动输出
        MOVAM   PD_DAT
KEY_SCAN ;; NOP             ;; 等待按键加载
        MOV     PA_DAT
        MOVAM   KEY_DAT,m
        MOV     AVAIL_KEY
        AND     KEY_DAT,m   ;; 可用按键数据PA[6~0]
        TMSC    KEY_DAT
        LGOTO   KEY_END
        BTSC    PD_DAT, 7   ;; 当PD[7]=1，按键扫描将会停止
        LGOTO   KEY_END
        RL      PD_DAT,m    ;; 改变PD数据PD[7]ßPD[6]ß …….PD[1]ßPD[0]
        LGOTO   KEY_SCAN
KRY_END …….
        MOV     STATUS_REG
        MOVAM   STATUS      ;; 取出STATUS
        MOV     ACC_REG     ;; 取出ACC
        IRET

        ORG     200h
INITIAL CLR     STATUS
        MOVLA   0xFF
        MOVAM   PA_DIR      ;; PA输入
        MOVLA   0xFF
        MOVAM   PD_PUD      ;; 下拉输入或pmos开漏输出
        MOVLA   0x00
        MOVAM   PD_DIR      ;; PD pmos开漏输出
        MOVAM   PD_DAT      ;; PD7~0浮动
```

```
        MOVLA   0x00        ;; PA6~0作为I/O输入工作
        MOVAM   PA_CTL
        MOVLA   0x7F
        MOVAM   WAKE_UP     ;; 脚位沿唤醒
        MOV     PA_DAT      ;; 等待脚位沿发生
        CLR     IRQF
        MOVLA   B'01000000' ;; 使能PORT A  irq
        MOVAM   IRQM
        BS      IRQM_CTL,7  ;; 使能IRQ
        NOP
LOOP    ……
        ……
        MOVLA   0x00        ;; 无按压KEY，PA6~0作为I/O输入工作
        MOVAM   PA_CTL
        MOVLA   0x7F
        MOVAM   WAKE_UP     ;; 脚位沿唤醒
        SLEEP               ;; 睡眠或暂停模式
        ……
        ……
        LGOTO   LOOP
```

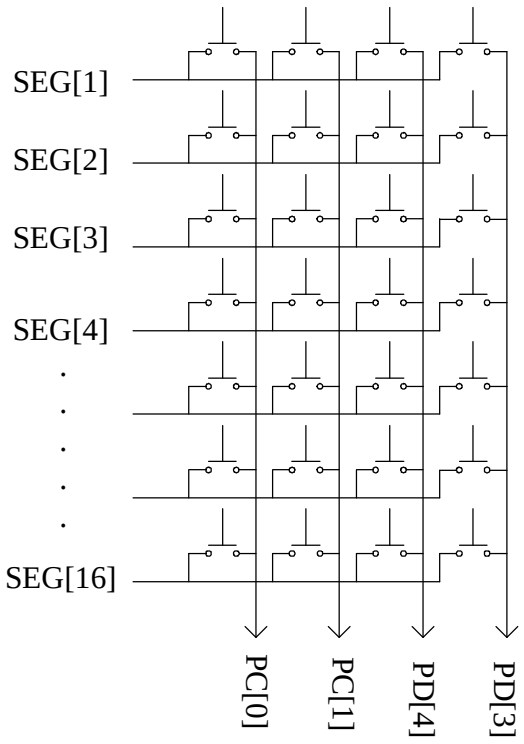


图.4.3.3　软件模式（PC[1:0]，PD[4:3]-输入，SEG-输出）

4.3.3-1 软件模式示例（PC[1:0] & PD[4:3]-输入，SEG[16:1]-输出）

```
#INCLUDE   "MK9A50P.INC"              ;; 仅正常
                                      ;; PC[1:0]或PD[4:3]输入带下拉（KI模式）
                                      ;; PC[7:0]可使用KI模式功能
                                      ;; SEG[16:1]开漏输出
#DEFINE    KEY_DATA    80h
#DEFINE    AVAIL_KEY   81h            ;; 可用按键（PA6~0）
#DEFINE    KEY_NUM     82h            ;; KEY扫描环路号码
           ORG         0x00
           LGOTO       INITIAL
ORG        0x04
           ……
           IRET

KEY        MOVLA       B'00010000'    ;; 软件读取按键 & 硬件模式1模式
           MOVAM       STROBE         ;; SEG[0]=hi 输出，其他浮动
                                      ;; PC[1:0] & PD[4:3]输入带下拉
KEY_SCAN   ;; NOP                     ;; 等待按键加载
           MOV         PC_DAT         ;; 读取PC[1:0]输入
           ;; MOV      PD_DAT         ;; 读取PD[4:3]输入
           MOVAM       KEY_DAT,m
           MOV         AVAIL_KEY
           AND         KEY_DAT,m      ;; 可用按键数据
           TMSC        KEY_DAT
           LGOTO       KEY_END
           MOV         KEY_NUM        ;; 按键扫描号码
           TMCOMPE     STROBE         ;; 比较STROBE=00011111
           LGOTO       KEY_END
           INC         STROBE,m
           LGOTO       KEY_SCAN
KRY_END    …….
           MOVLA       B'00000000'    ;; 使能硬件按键扫描
           MOVAM       STROBE         ;; PA[6~0]下拉 & 输入缓冲关闭
                                      ;; 按KEY有无功率消耗
           RET

           ORG         100h           ;; 主程序，按KEY有无功率消耗
INITIAL    CLR         STATUS
           MOVLA       B'00011111'    ;; KEY号码 =16（SEG16~1）
```

```
        MOVAM   KEY_NUM
        MOVLA   B'00000011'   ;; PC[1:0]
        ;; MOVLA B'00011000'  ;; PD[4:3]
        MOVAM   AVAIL_KEY
        MOVLA   B'00000011'   ;; 5 X COM，PC[1]=COM[6]，PC[0]=COM[7]
        MOVAM   LBASDT
        MOVLA   B'11100111'   ;; PD[3]=SEG[23]，PD[4]=SEG[24]
        MOVAM   PAD_CTL1
        MOVLA   B'11111111'   ;; PC & PD作为输入模式工作
        MOVAM   PC_DIR
        MOVAM   PD_DIR
        MOVLA   B'00000011'   ;; PC[1:0]作为KI输入工作，低功率
        MOVAM   PC_CTL
        MOVLA   B'00011000'   ;; PD[4:3]作为KI输入工作，低功率
        MOVAM   PD_CTL
        NOP
LOOP    ……                    ;; 主环路，PC[1:0] & PD[4:3]是输入浮动 &
        ……                    ;; 输入缓冲关闭，无功率消耗
        MOVLA   B'00100000'   ;; 软件模式 & SEG[16:1]=FFFFh输出 &
                              PC[1:0] & PD[4:3]输入缓冲开启及下拉使能
        MOVAM   STROBE
        MOV     PC_DAT
        ;; MOV  PD_DAT
        MOVAM   KEY_DAT
        MOV     AVAIL_KEY
        AND     KEY_DAT
        BTSC    KEY_DAT
        LCALL   KEY
        MOVLA   B'00000000'   ;; Port C输入缓冲关闭 & 下拉禁止
        MOVAM   STROBE        ;;
        ……
        ……
        LGOTO   LOOP
```

4.3.3-2 软件模式示例（PC[2:0] & PD[7:0]-输入，SEG[16:1]-输出）

```
#INCLUDE   "MK9A50P.INC"          ;; 仅正常
                                  ;; PC[2:0]或PD[7:0]输入带下拉（I/O模式）
                                  ;; SEG[16:1]开漏输出
#DEFINE    KEY_DATA   80h
```

```
#DEFINE     AVAIL_KEY   81h             ;; 可用按键（PA6~0）
#DEFINE     KEY_NUM     82h             ;; KEY扫描环路号码
            ORG         0x00
            LGOTO       INITIAL
ORG         0x04
            ……
            IRET

KEY         MOVLA       B’00010000’     ;; 软件读取按键 & 硬件模式1模式
            MOVAM       STROBE          ;; SEG[0]=hi 输出，其他浮动
KEY_SCAN    ;; NOP                      ;; 等待按键加载
            MOV         PC_DAT          ;; 读取PC[2:0]输入
            ;; MOV      PD_DAT          ;; 读取PD[7:0]输入
            MOVAM       KEY_DAT,m
            MOV         AVAIL_KEY
            AND         KEY_DAT,m       ;; 可用按键数据
            TMSC        KEY_DAT
            LGOTO       KEY_END
            MOV         KEY_NUM         ;; 按键扫描号码
            TMCOMPE     STROBE          ;; 比较STROBE=00011111
            LGOTO       KEY_END
            INC         STROBE,m
            LGOTO       KEY_SCAN
KRY_END     …….
            RET

            ORG         100h            ;; 主程序，按KEY有无功率消耗
INITIAL     CLR         STATUS
            MOVLA       B’00011111’     ;; KEY号码 =16（SEG16~1）
            MOVAM       KEY_NUM
            MOVLA       B’00000111’     ;; PC[2:0]
            ;; MOVLA    B’11111111’     ;; PD[7:0]
            MOVAM       AVAIL_KEY
            MOVLA       B’00000010’     ;; 4 X COM，PC[2]=COM[5]，PC[1]=COM[6]，
                                           PC[0]=COM[7]
            MOVAM       LBASDT
            MOVLA       B’00000000’     ;; PD[7:0]作为I/O工作
            MOVAM       PAD_CTL1
            MOVAM       PC_DAT
```

```
        MOVAM   PD_DAT
        MOVLA   B'11111111'
        MOVAM   PC_PUD
        MOVAM   PD_PUD
        MOVLA   B'00000000'   ;; PC & PD作为pmos开漏输出模式工作
                                 & 输出浮动
                                 & 输入缓冲器关闭，无功率消耗
        MOVAM   PC_DIR
        MOVAM   PD_DIR
        NOP
LOOP    ……
        MOVLA   B'00100000'   ;; 软件模式 & SEG[16:1]=FFFFh输出
        MOVAM   STROBE
        MOVLA   B'11111111'
        MOVAM   PC_DIR
        ;; MOVAM   PD_DIR
        MOV     PC_DAT
        ;; MOV     PD_DAT
        MOVAM   KEY_DAT
        MOV     AVAIL_KEY
        AND     KEY_DAT
        BTSC    KEY_DAT
        LCALL   KEY
        MOVLA   B'00000000'   ;; 低功率消耗
        MOVAM   PC_DIR        ;; 输出浮动（pmos开漏 + 低输出）
        ;;MOVAM  PD_DIR
        MOVLA   B'00000000'   ;; 使能硬件按键扫描
        MOVAM   STROBE
        ……
        ……
        LGOTO   LOOP
```

4.4. 中断 & 暂停释放

MK9A50P提供8种中断事件。IRQM及IRQF寄存器用于控制或宣布所有中断的请求状态。外部中断可通过PA0~7的一个从高到低的转换信号触发，相关中断请求标记（PAF，IRQF的bit7）将被置1。IRQM用于使能/禁止中断，IRQF用于指示哪一种中断发生。如果特定IRQM未使能，则硬件中断将不会发生。但不管IRQM使能或禁止，IRQF都将反应状态。例如，用户使能TM1去开始读数。如果IRQM的bit 1使能，当定时器溢出硬件中断将发生，IRQF的bit 1将被置。与此同时，程序将跳到中断向量。用户应在中断服务例行程序清除IRQF，否则中断将不会完全工作。另一种情况是如果IRQM的bit 1禁止，当定时器溢出中断将不会发生，但IRQF的bit 1仍将被设置。程序将不会跳到中断向量。

IRQM_CTL ($2Fh)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRQM_CTL	INTM	--	--	--	--	--	--	--

- Bit7（INTM）：通用使能/禁止位
  - 0：禁止，所有中断屏蔽
  - 1：使能，所有中断不屏蔽

<注> 当中断正在运行时，INTM 将复位到“0”以防止其他中断发生。运行结束后，RETI 指令将设置 RETI 为“1”。

IRQM ($31h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRQM	--	PACM	PINTM	2HZM	PHM	TM3M/ PWM3F/ CAPT3M/ RFC3M	TM2M/ PWM2F/ CAPT2M/ RFC2M	TM0M/ TONEM

- Bit7（INTM）：通用使能/禁止位
  - 0：禁止，所有中断屏蔽
  - 1：使能，所有中断不屏蔽

CPU_RESUME 1 ($30h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CPU_RESUME1	--	PACR	PINTR	2HZR	PHR	TM3R/ PWM3R/ CAPT3R/ RFC3R	TM2R/ PWM2R/ CAPT2R/ RFC2R	TM0R/ TONER

- Bit7~0：暂停释放模式控制 1
  - 0：禁止
  - 1：使能

CPU_RESUME	IRQM	中断	中断标记	正常模式	HALT模式	SLEEP模式
1	X	禁止	V	下一个命令	1. 唤醒系统时钟 2. 下一条指令	不能使用
0	1	V	V	跳到004h	1. 唤醒系统时钟 2. LCALL 004h（进入IRQ）	1. 唤醒系统时钟 2. LCALL 004h（进入IRQ）
0	0	X	V	X	无IRQ & 无唤醒功能	无IRQ & 无唤醒功能

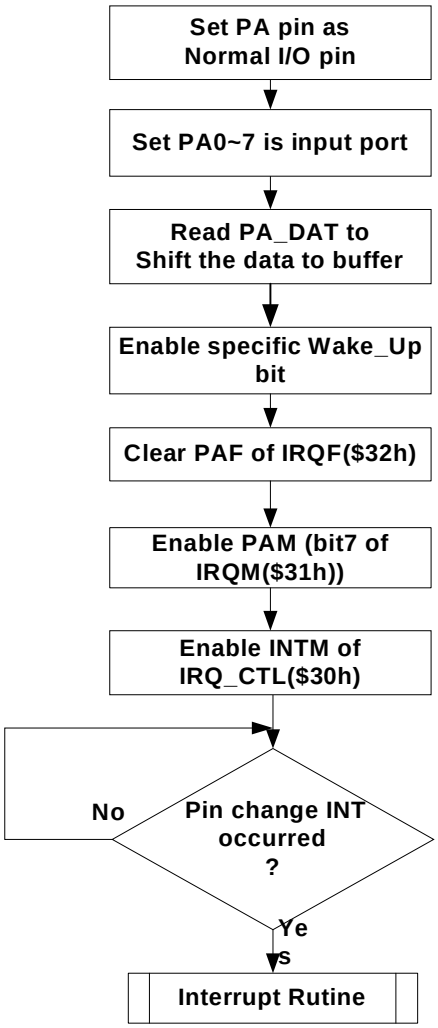
IRQF ($32h)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRQF	--	PACF	INTF	2HZF	PHR	TM3F/ PWM3F/ CAPTF/ RFC3F	TM2F/ PWM2F/ CAPT2F/ RFC2F	TM0F/ TONEF

- Bit6（PACF）：PA0~7 及 PC~70 中断请求标记
  - 0：PA 及 PC 中断请求关闭
  - 1：PA 及 PC 中断请求开启
- Bit5（INTF）：INT 脚位中断请求标记
  - 0：INT脚位（PD5）中断请求关闭
  - 1：INT脚位（PD5）中断请求开启
- Bit4（2HZF）：2HZ 中断请求标记
  - 0：2HZF溢出中断请求关闭
  - 1：2HZF溢出中断请求开启
- Bit3（PHF）：TM4 中断请求标记
  - 0：PH溢出中断请求关闭
  - 1：PH溢出中断请求开启
- Bit2（TM3F/CAPTF/PWMF/RFCF）：TM3 中断请求标记
  - 0：TM2 溢出中断请求关闭
  - 1：TM2溢出中断请求开启
- Bit1（TM2F/CAPTF/ PWMF/RFCF）：TM2 中断请求标记
  - 0：TM2 溢出中断请求关闭
  - 1：TM2 溢出中断请求开启
- Bit0（TM0F/TONEF）：TM0/捕捉中断标记
  - 0：TM0 溢出或 TONE 中断请求关闭
  - 1：TM0 溢出或 TONE 中断请求开启

4.5. 外部中断脚位– PA[0~7]，PC[0~7] & PD[5]

A 口（PA）和 C 口（PC）提供外部中断及唤醒功能。当器件未在睡眠模式下，PA 输入信号将作为外部中断服务。当中断发生，程序将跳到 004H（中断向量）。如果器件在睡眠模式下，PA 输入信号将作为唤醒功能服务。当唤醒信号输入，器件将首先让系统时钟工作然后等待唤醒定时器（通过 WDT_CTL 寄存器$3Bh 设置）溢出，在那之后，程序将跳到 004H。以下流程图描述如何设置 A 口作为外部中断或唤醒功能工作。



4.6. 电阻至频率转换器（RFC）

RFC 是一种单斜率积分电路，可用于像低速 16 位 ADC 去检测一些电阻器类型的传感器。MK9A50P 有两套 RFC 通道，通过设置寄存器可带许多不同连接。

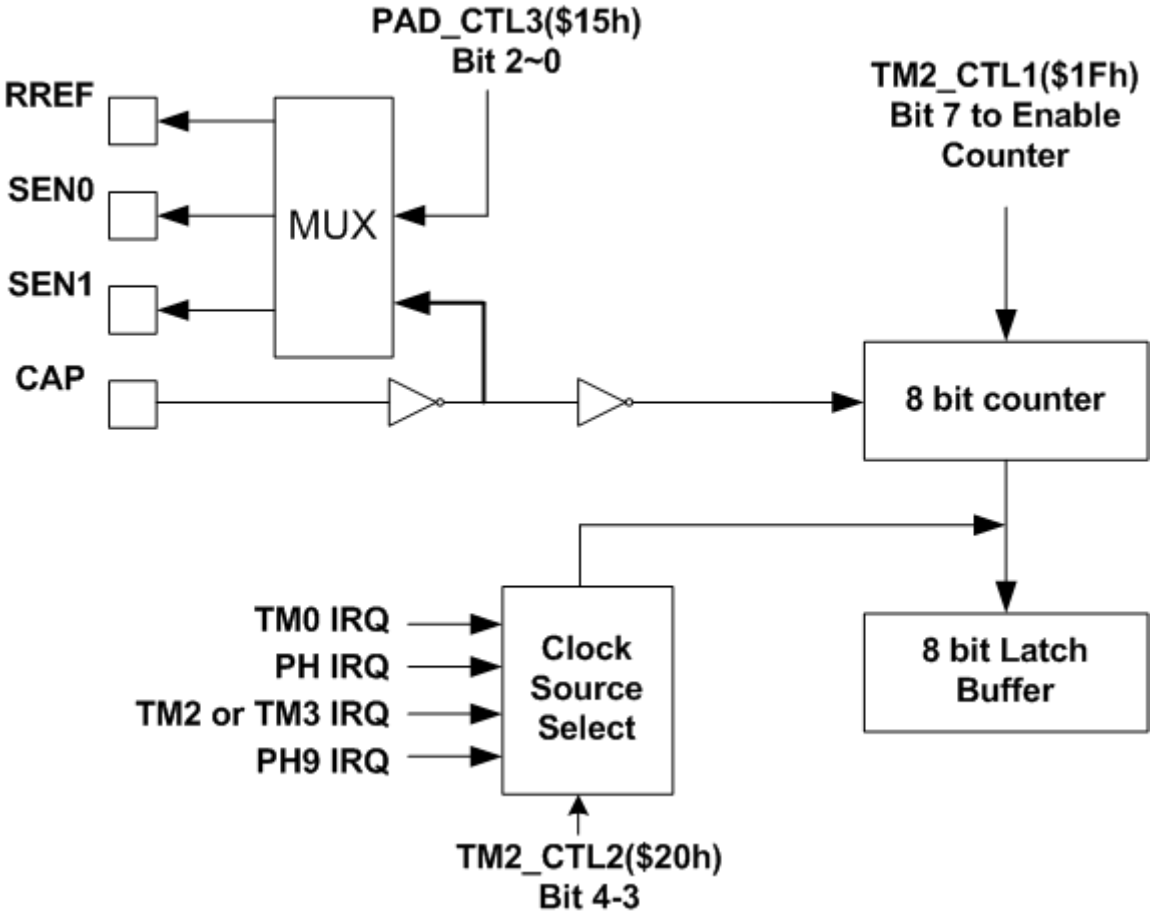


图.4.6.1 RFC 应用示意图

一套 RFC 区块由四个外部脚位组成，分别是：

(a) RREF：参考电阻器输出脚位
(b) SEN0：传感器 1 输出脚位
(c) SEN1：传感器 2 输出脚位
(d) CAP：振荡输入脚位

因为所有的 RFC 脚位都与 COM7~10 及（PB[2:4]，PC[2]）共享，使用者应首先设置特定寄存器如下：

(a) 设置 LBASDT（$33h）的 bit2~0 为（1，0，0）。它意味着设置为 1/6 占空比，然后 COM7~10 为 I/O 口。
(b) 设置 PAD_CTL1（$13h）所有的 Bit3~0 为“0”，意味着设置所有口作为 RFC 功能。

4.6.1 定时器 2 作为 8 位捕捉动作

这是 RFC 16 位定时器功能之一。16 位计数器源通过设置 TM2_CTL1（$1Fh）bit1~0 可以是 FCLK，PH0X2，PH4 或 PH_CLK。CAPn 输入时钟将成为锁存计数器到锁存缓冲器事件。

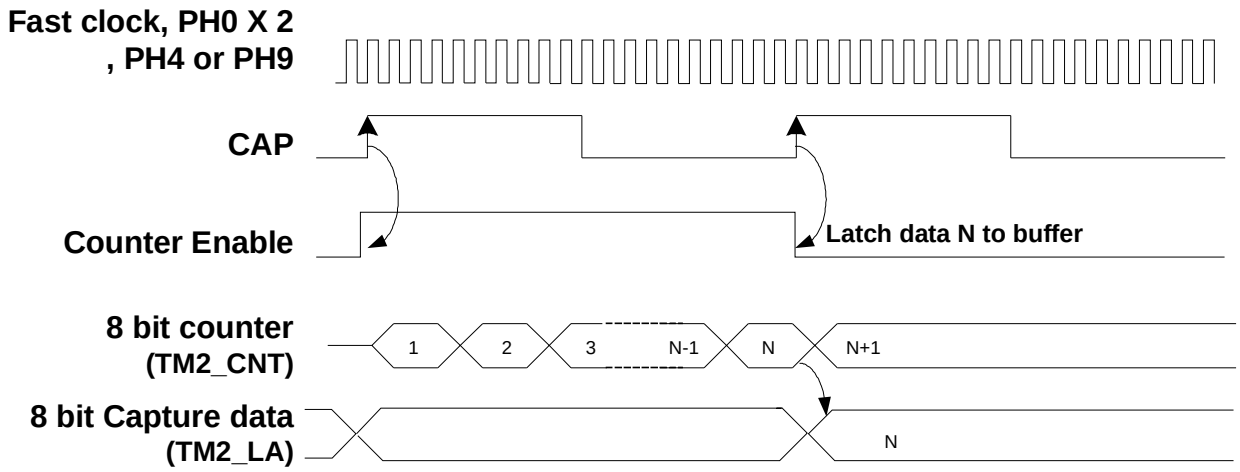


图.4.6.1 RFC 定时器作为 8 位捕捉动作的时间图

4.6.2 定时器2作为RFC计数器动作

当 RFC 作为 16 位事件计数器动作去计算外部 RC 振荡频率时，当定时器溢出，TM1 将成为时基，触发信号去锁存数据到缓冲器。时间图及设置流程如下：

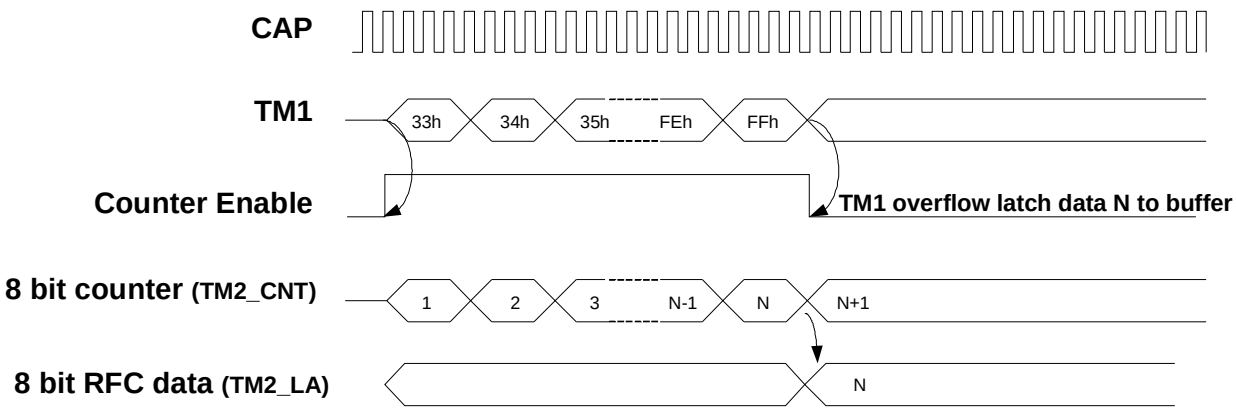


图.4.6.2 RFC 定时器作为 RFC 计数动作的时间图

4.7. EL 面板驱动功能

MK9A50P 为 LCD 逆光提供 EL 面板驱动。应用电路如图.4.7.1。因为 ELP/ELC 脚位与 SEG39~40 共享，使用者可首先通过设置 SEG_CTL1 ($13h)选择脚位定义。

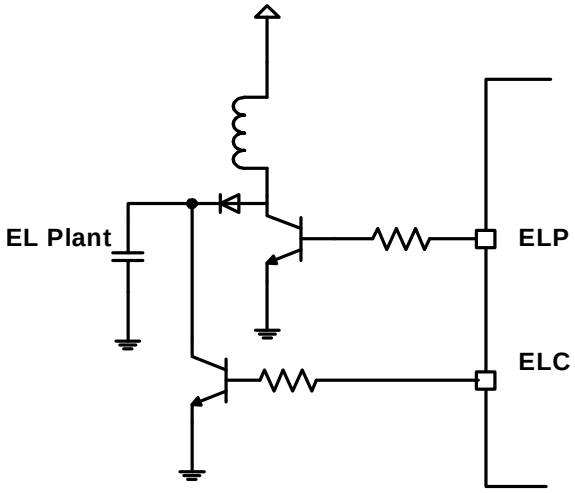


图.4.7.1 EL 面板连接的应用电路

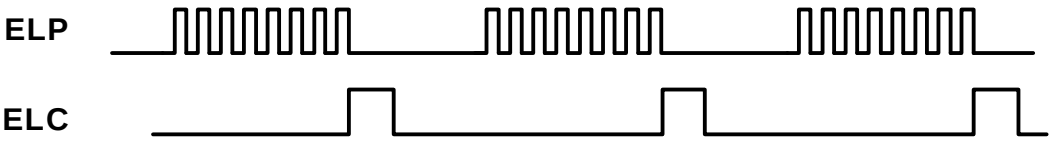


图.4.7.2 EL 驱动时间图

PH_CTL ($36h) bit6~7 用于控制 EL 驱动。Bit6 用于设置 ELP/ELC 频率及占空比，见下表。Bit7 用于开启/关闭 EL 驱动电路。

Bit	符号	描 述	
6	EL_SEL	1：高速	ELP，ELC：高速模式 如果高速时钟=500Khz ELP：21.8KHz，15/16占空比 ELC：683Hz，1/4占空比（L:H=3:1）
		0：低速	ELP，ELC：低速模式 如果低速时钟=32Khz ELP：16KHz，3/4占空比 ELC：512Hz，1/4占空比（L:H=3:1）

4.8. 低电压复位（LVR）

LVR 功能用于当功率降低时在未知状况下引起逻辑以预防系统故障。用户可通过设置配置寄存器来设置不同电压或不使用。不同批次芯片及环境下，下表中的电压允许有少许误差。

Bit7	Bit6	检测电压
LV1	LV0	
0	0	1.5V
0	1	1.7V
1	0	2.0V
1	1	不使用

4.9. 低电压检测（LVD）

LVD 功能用于检测电池低压时提醒用户去更换电池。用户可通过 SYS_CTL ($3Eh) bit4~3 设置检测电压及状态。不同批次芯片及环境下，下表中的电压允许有少许误差。

SYS_CTL ($3Eh)

Bit	符号	描 述	
4~3	LVD1~0	低电压检测器	
		1 1	ON（2.56V）
		1 0	ON（2.40V）
		0 1	ON（2.68V）
		0 0	功能关闭
2	LV	低电压检测输出（仅只读）	
		1	VDD < 2.56V（2.58V或2.40V）
		0	VDD > 2.56V（2.68V或2.40V）

4.10. 其他寄存器

PH_CTL ($36h):

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PH_CTL	ELON	EL_SEL	EL_P	CLR	PH_I1	PH_I0	PH_S1	PH_S0

- Bit7：此位是用于开启/关闭 EL 光电荷 pump 波形
  - 1：ELP/ELC 开启
  - 0：ELP/ELC 关闭，输出信号为低
- Bit6：EL 光电荷 pump 波形控制

Bit	符号	描 述	
6	EL_SEL	1：高速	ELP，ELC：高速模式 如果高速时钟=500Khz ELP：15.6KHz，15/16占空比 ELC：683Hz，1/4占空比（L:H=3:1）
		0：低速	ELP，ELC：低速模式 如果低速时钟=32Khz ELP：16KHz，3/4占空比 ELC：512Hz，1/4占空比（L:H=3:1）

- Bit5：EL 焊盘控制

位\数值	11	10	01	00
(PH_CTL.EL_P),B6	Don't	ELP	SEG39	PD6
(PH_CTL.EL_C),B7	Don't	ELC	SEG40	PD7

- Bit4：清除驱动 PH11~PH15

  0：不清除

  1：清除 PH11~PH15 及自动清除 BIT5（CLR）到“0”

- Bit3~2：PH IRQ（PHF）源选择

Bit	符号	描 述	
3~2	PH_I1~0	PH_I1~0	PH IRQ源选择
		0 0	PHR <= PH10 (32hz)
		0 1	PHR <= PH11 (16hz)
		1 0	PHR <= PH12 (8hz)
		1 1	PHR <= PH13 (4hz)

- Bit1~0：（PH_CLK）PH 源选择

Bit	符号	描 述	
1~0	PH_S1~0	PH_S1~0	PH_CLK
		0 0	PH_CLK <= PH9 (64hz)
		0 1	PH_CLK <= PH7 (256hz)
		1 0	PH_CLK <= PH8 (128hz)
		1 1	PH_CLK <= PH10 (32hz)

4.10.1 PH IRQ & 暂停模式示例

```
#INCLUDE   "MK9A50P.INC"          ；暂停模式，PH=PH10唤醒
ORG        0x00
           LGOTO     INITIAL

           ORG       004
           MOVLA     ‘01110111’
           MOVAM     IRQF          ; 清除PH中断标记
           ……..
           IRETI

           ORG       0x20
INITIAL
           CLR       PA_DAT        ；清除浮动
           CLR       PD_DAT        ；清除浮动
           MOVLA     0x00          ; 设置PA为输出脚位
           MOVAM     PA_DIR
           MOVLA     0xFF
           MOVAM     PA_PUD1       ; 设置PA为正常输出脚位
           MOVAM     PA_PUD2
           MOVAM     PD_PUD1       ; 设置PD为正常输出脚位
           MOVAM     PD_PUD2
           MOVLA     B'01000111'   ; 帧=42hz，com1~10
           MOVAM     LBASDT
           MOVLA     B'00110010'   ; b5.4=11，低功率，LCD开启
           MOVAM     LCD_CTL
           CLR       IRQF          ; 清除中断标记
           MOVLA     B'00001000'   ; 设置2HZM中断
           MOVAM     IRQM
           BS        IRQM,7        ; 使能中断
LOOP
           BC        LCD_CTL,1     ；关闭LCD
           BS        SYS_CTL,6     ; 设置HALT；OSC活动但CPU关闭
                                   ; 如果唤醒，仅系统时钟被开启
           BS        LCD_CTL,1     ；开启LCD
           …….
           LGOTO     LOOP
END
```

5. LCD 驱动器

MK9A50P 最大值为 900 分段（10com * 42seg）。有 1/4，1/5，1/6，1/7，1/8 及 1/10，1/18 占空比可选择。有两种功率模式，Li 和 EXT。许多 COM 脚位可与 I/O 口或一些特定功能分享以便让它更灵活。如果使用较少的 COM 脚位，则剩余 COM 脚位可设置为其他功能。

5.1. LCD 焊盘连接

MK9A50P 有不同种类的功率模式，占空比，偏压组合。不同偏压的焊盘连接是不一样的，见下图：

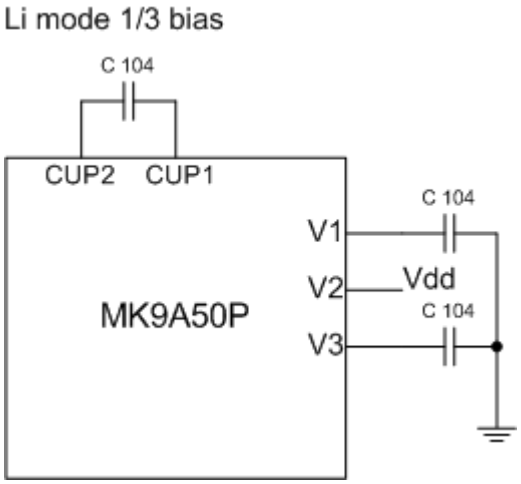


图.5.1.1　不同模式下的 LCD 焊盘连接

5.2. LCD 属性设置

有许多寄存器去设置 LCD 驱动器属性，如下：

5.2.1 偏压设置

偏压设置在配置寄存器阶段。一旦它是固定的，用户不能通过软件来改变它。只有一种偏压可供选择：1/3偏压。

5.2.2 占空比（COM）及帧频率设置

LBASDT（$33h）用于设置占空比及LCD帧频率。一旦占空比被设置，意味着选择脚位将被作为LCD COM输出使用。剩余的COM脚位将自动成为一个I/O口。寄存器定义如下：

LBASDT($33h)：LCD 占比选择（R/W）

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LBASDT	LCD1	LCD0	FRAM1	FRAM0	LDUTY3	LDUTY2	LDUTY1	LDTUY0

- Bit 6：LCD0模式选择

  1 1：LED模式1 – COMn是高态，SEGn是低态

  1 0：LED模式2 – COMn是低态，SEGn是低态

  0 1：LED模式3 – COMn是低态，SEGn是高态

  0 0：LI模式 – 1/3偏压电荷泵

DUTY	LCD ON		LCD OFF	
	COMn	SEGn	COMn	SEGn
LCD	4.5V / 0V	0V / 4.5V	0	0
LED1	VDD	0	0	0
LED2	0	0	VDD	0
LED3	0	VDD	0	0

	LCD OFF	ALL ON	LCD ON	ALL OFF
LCD	V	V	V	V
LED1	V	X	V	X
LED2	V	X	V	X
LED3	V	X	V	X

- Bit5~4：LCD 帧控制（仅写入）

  0 0：FRAME 1

  0 1：FRAME 2

  1 0：FRAME 3

  1 1：FRAME 4

DUTY	LCD FRAME (Hz)			
	FRAME 1	FRAME 2	FRAME 3	FRAME 4
1/4	64.05	42.67	85.33	128.10
1/5	64.68	43.12	86.23	129.36
1/6	64.05	42.67	85.33	128.10
1/7	64.05	42.67	85.33	128.10

DUTY	LED FRAME (Hz)			
	FRAME 1	FRAME 2	FRAME 3	FRAME 4
1/4	128.10	85.33	170.66	256.20
1/5	129.36	86.24	172.46	258.72
1/6	128.10	85.33	170.66	256.20
1/7	128.10	85.33	170.66	256.20

<注> 此频率表格是基于 32KHz 源时钟。如果使用者仅选择快时钟模式，而时钟不是 32KHz，则频率将与上表不一样。

- Bit3~0：LCD 驱动占空比设置（R/W）

LDUTY3	LDUTY2	LDUTY1	LDUTY0	DUTY	COM 1~6	
					COM is for LCD	COM to be I/O
0	0	1	0	1/4	1~4	5~8
0	0	1	1	1/5	1~5	6~8
0	1	0	0	1/6	1~6	7~8
0	1	0	1	1/7	1~7	8
0	1	1	0	1/8	1~-8	X
Others				Reserved		

5.2.3 LCD pump 频率及ON/OFF控制

LCD_CTL($35h)：LCD 控制（R/W）（缺省= 0000xx00b）

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LCD_CTL	PUMP1	PUMP0	POW1	POW0	OVP1	OVP0	LCDM1	LCDM0

- Bit7~6（PUMP1~0）：LCD 电荷 pump 时钟选择
  - 00：原始 pump 时钟
  - 01：原始 pump 时钟 x 2
  - 10：未使用
  - 11：原始 pump 时钟 x 4
- Bit5~4（POW1~0）：低速省电控制（暂停模式）
  - 00：缺省
  - 11：低功率（建议，暂停模式时低功率）
- Bit3~2（OVP1~0）：LCD 波形控制
  - 00：重叠
  - 01：$T_{NON\text{-}OVERLAP}$ （15uS）
  - 10：$TB_{NON\text{-}OVERLAP\ B}$X 2 （30uS）
  - 11：$TB_{NON\text{-}OVERLAP\ B}$X 4 （60uS）
- Bit1~0（LCDM1~0）：LCD/LED 模式选择（默认：ALL OFF 模式）

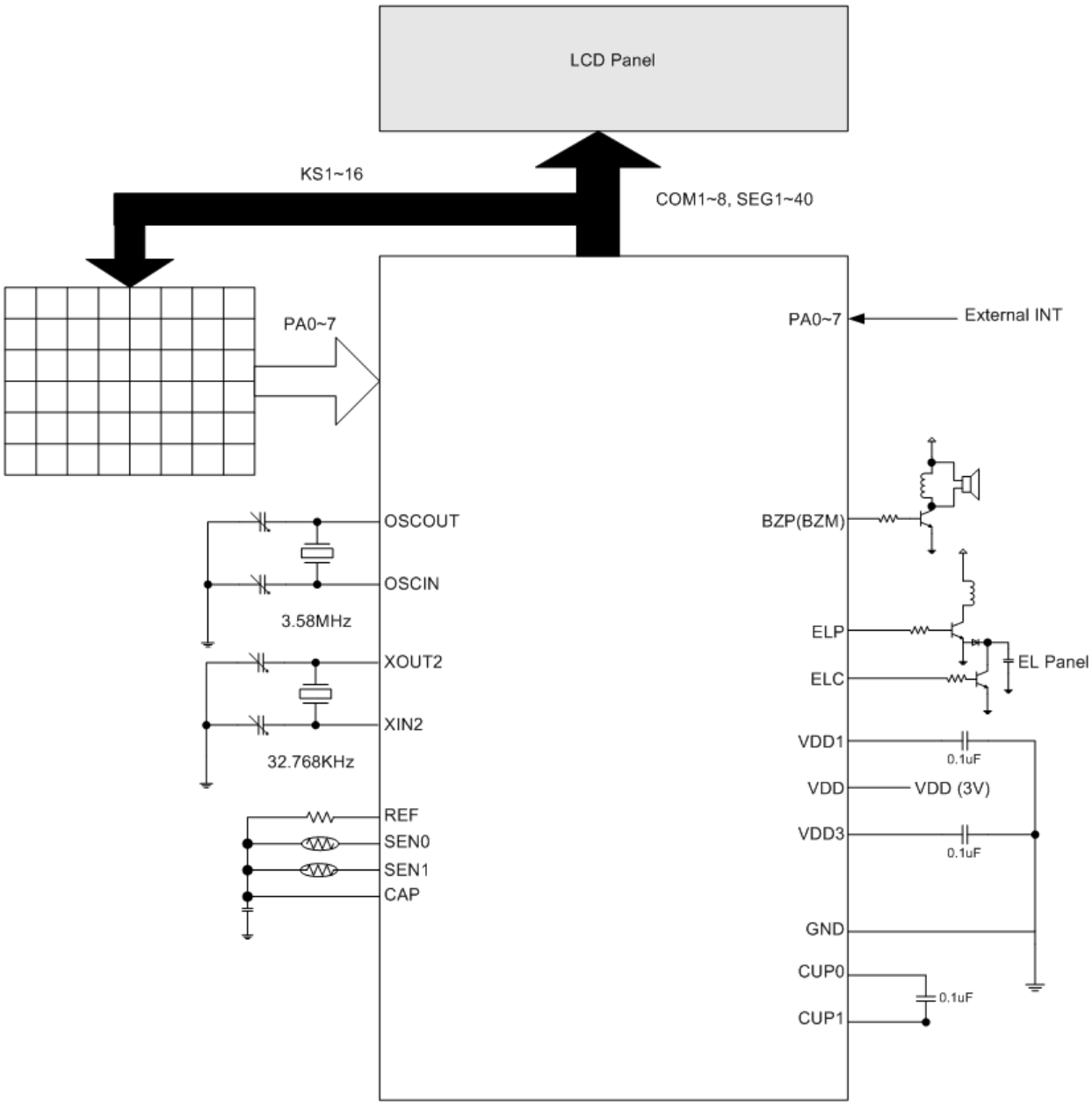
Bit1	Bit0	模式	状 态
0	0	正常模式	LCD 电源 OFF（LED OFF）
0	1	ALL ON 模式	当 ALL ON 设置为“0”，所有分段驱动开启波形
1	0	正常模式	1. 寄存器的显示数据是输出到分段驱动 LCD ON 2. LED ON
1	1	ALL OFF 模式	当 ALL OFF 设置为“1”，所有分段驱动关闭波形

5.3. LCD 显示 RAM 映射

LCD 驱动有一个 LCD RAM。每一位 RAM 映射到特定的 COM/分段见下表。如果 LCD RAM 不使用，他们可被用作工作 RAM 来存储数据。

	Bit0~Bit7		Bit0~Bit7
	C1~C8		C1~C8
SEG1	$40	SEG22	$55
SEG2	$41	SEG23	$56
SEG3	$42	SEG24	$57
SEG4	$43	SEG25	$58
SEG5	$44	SEG26	$59
SEG6	$45	SEG27	$5A
SEG7	$46	SEG28	$5B
SEG8	$47	SEG29	$5C
SEG9	$48	SEG30	$5D
SEG10	$49	SEG31	$5E
SEG11	$4A	SEG32	$5F
SEG12	$4B	SEG33	$60
SEG13	$4C	SEG34	$61
SEG14	$4D	SEG35	$62
SEG15	$4E	SEG36	$63
SEG16	$4F	SEG37	$64
SEG17	$50	SEG38	$65
SEG18	$51	SEG39	$66
SEG19	$52	SEG40	$67
SEG20	$53	SEG41	$68
SEG21	$54	SEG42	$69

6. 典型应用电路



Application circuit

7. 指令表

JUMP INSTRUCTION				
LCALL I	Call subroutine. However, LCALL can addressing 16K address	2	None	01ii iiii iiii iiii
LGOTO I	Go branch to any address	2	None	00ii iiii iiii iiii
LOGIC				
AND M, a	(M) · (acc) → (acc)	1	Z	1010 1000 MMMM MMMM
AND M, m	(M) · (acc) → (M)	1	Z	1010 1001 MMMM MMMM
ANDLA I	Immediate · (acc) → (acc)	1	Z	1111 1000 iiii iiii
COM M, a	~(M) → (acc)	1	Z	1010 0100 MMMM MMMM
COM M, m	~(M) → (M)	1	Z	1010 0101 MMMM MMMM
IOR M, a	(M) or (acc) → (acc)	1	Z	1011 1110 MMMM MMMM
IOR M, m	(M) or (acc) → (M)	1	Z	1011 1111 MMMM MMMM
IORLA I	Immediate or (acc) → (acc)	1	Z	1111 0010 iiii iiii
RL M, a	Rotate left from m to acc m[6:0] → acc[7:1] m[7] → acc[0]	1	None	1110 0000 MMMM MMMM
RL M, m	Rotate left from m to itself m[6:0] → m[7:1] m[7] → m[0]	1	None	1110 0001 MMMM MMMM
RLC M, a	Rotate left from m to acc m[7] → c m[6:0] → acc[7:1] c → acc[0]	1	C	1110 0010 MMMM MMMM
RLC M, m	Rotate left from m to itself m[7] → c m[6:0] → m[7:1] c → m[0]	1	C	1110 0011 MMMM MMMM
SL0 M, a	Shift left from m to acc m[6:0] → acc[7:1] 0 → acc[0]	1	None	1110 0100 MMMM MMMM
SL0 M, m	Rotate left from m to itself m[6:0] → m[7:1] 0 → m[0]	1	None	1110 0101 MMMM MMMM
SL1 M, a	Shift left from m to acc m[6:0] → acc[7:1] & 1 → acc[0]	1	None	1110 0110 MMMM MMMM
SL1 M, m	Rotate left from m to itself m[6:0] → m[7:1] & 1 → m[0]	1	None	1110 0111 MMMM MMMM
RR M, a	Rotate right from m to acc 0 → acc[7]	1	None	1110 1000 MMMM MMMM

	m[7:1] → acc[6:0]			
RR M, m	Rotate right from m to itself M[0] → m[7] m[7:1] → m[6:0]	1	None	1110 1001 MMMM MMMM
RRC M, a	Rotate right from m to acc m[0] → c , c → acc[7] m[7:1] → acc[6:0]	1	C	1110 1010 MMMM MMMM
RRC M, m	Rotate right from m to itself m[0] → c , c → m[7] m[7:1] → m[6:0]	1	C	1110 1011 MMMM MMMM
SR0 M, a	Rotate right from m to acc 0 → acc[7] m[7:1] → acc[6:0]	1	None	1110 1100 MMMM MMMM
SR0 M, m	Rotate right from m to itself 0 → m[7] m[7:1] → m[6:0]	1	None	1110 1101 MMMM MMMM
SR1 M, a	Rotate right from m to acc 1 → acc[7] m[7:1] → acc[6:0]	1	None	1110 1110 MMMM MMMM
SR1 M, m	Rotate right from m to itself 1 → m[7] m[7:1] → m[6:0]	1	None	1110 1111 MMMM MMMM
SWAP M, a	m[7:4] → acc[3:0] m[3:0] → acc[7:4]	1	None	1011 1100 MMMM MMMM
SWAP M, m	m[7:4] ←→ m[3:0]	1	None	1011 1101 MMMM MMMM
XOR M, a	(M) xor (acc) → (acc)	1	Z	1011 0110 MMMM MMMM
XOR M, m	(M) xor (acc) → (M)	1	Z	1011 0111 MMMM MMMM
XORLA I	Immediate xor (acc) → (acc)	1	Z	1111 1001 iiii iiii
MATHEMATICS				
ADD M, a	(M)+(acc) → (acc)	1	C, DC, Z	1010 1010 MMMM MMMM
ADD M, m	(M)+(acc) → (M)	1	C, DC, Z	1010 1011 MMMM MMMM
ADDC M,a	(M)+(acc) + (carry) → (acc)	1	C, DC, Z	1011 1010 MMMM MMMM
ADDC M,m	(M)+(acc) + (carry) → (M)	1	C, DC, Z	1011 1011 MMMM MMMM
ADDLA I	Immediate + (acc) → (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)+ (carry) → (acc)	1	C, DC, Z	1101 0100 MMMM MMMM
SUBC M, m	(M)+(/acc) + (carry) → (M)	1	C, DC, Z	1101 0101 MMMM MMMM
DAA M, a	Decimal Adjust M to ACC If ACC[3:0] > 9 or DC=1 Then ACC[3:0] ← ACC[3:0]+6, DC1=~DC else ACC[3:0] ← ACC[3:0], DC1=0 If ACC[7:4]+DC1 > 9 or C=1 Then ACC[7:4] ← ACC[7:4]+6+DC1, C=1 else ACC[7:4] ← ACC[7:4]+DC1, C=C	1	C	1101 0110 MMMM MMMM
DAA M, m	Decimal Adjust M to memory If ACC[3:0] > 9 or DC=1 Then M[3:0] ← ACC[3:0]+6, DC1=~DC else M[3:0] ← ACC[3:0], DC1=0 If ACC[7:4]+DC1 > 9 or C=1 Then M[7:4] ← ACC[7:4]+6+DC1, C=1 else M[7:4] ← ACC[7:4]+DC1, C=C	1	C	1101 0111 MMMM MMMM

DAS M, a	Decimal Adjust M to ACC
	If ACC[3:0] > 9 or DC=0
	Then
	ACC[3:0] ← ACC[3:0]-6,
	DC1=DC
	Else
	ACC[3:0] ← ACC[3:0], DC1=0
	If ACC[7:4] –DC1> 9 or C=0
	Then
	ACC[7:4] ← ACC[7:4]-6-DC1,
	C=0
	else
	ACC[7:4] ← ACC[7:4]-DC1,
	C=~C
		1	C	1101 1110 MMMM MMMM

DAS M, m	Decimal Adjust M to memory If ACC[3:0] > 9 or DC=0 Then M[3:0] ← ACC[3:0]-6, DC1=DC else M[3:0] ← ACC[3:0], DC1=0 If ACC[7:4]-DC1 > 9 or C=0 Then M[7:4] ← ACC[7:4]-6-DC1, C=1 else M[7:4] ← ACC[7:4]-DC1, C=C	1	C	1101 1111 MMMM MMMM

DEC M, a	(M) - 1 → (acc)	1	Z	1010 1100 MMMM MMMM
DEC M, m	(M) - 1 → (M)	1	Z	1010 1101 MMMM MMMM
INC M, a	(M) + 1 → (acc)	1	Z	1011 0000 MMMM MMMM
INC M, m	(M) + 1 → (M)	1	Z	1011 0001 MMMM MMMM
MOVAM m	(acc) → (M)	1	None	1010 0001 MMMM MMMM
MOV M, a	(M) → (acc)	1	Z	1010 0110 MMMM MMMM
MOV M, m	(M) → (M)	1	Z	1010 0111 MMMM MMMM
MOV2 M, a	(M) → (acc)	1	None	1111 0110 MMMM MMMM
MOV2 M, m	(M) → (M)	1	None	1111 0111 MMMM MMMM
MOVLA I	Immediate data → acc	1	None	1111 0000 iiii iiii
SUBLA I	(immediate data)-(Acc) → (Acc)	1	C, DC, Z	1111 0100 iiii iiii
SUB M, m	(M)–(acc) → (M)	1	C, DC, Z	1011 0101 MMMM MMMM
SUB M, a	(M) –(acc) → (acc)	1	C, DC, Z	1011 0100 MMMM MMMM
OTHER OPERATION				
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
IRETI	Return and enable INTM(for IRQ)	2	None	1111 1111 1111 0010
IRET	Return (for IRQ)	2	None	1111 1111 1111 0011
RETLA	Return & Immediate data → acc	2	None	1101 1100 iiii iiii
SLEEP	Enter sleep (saving) mode	1	$\overline{TO}$, $\overline{PD}$	1111 1111 1111 0100
CONDITION OPERATION				
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 → (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	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
TMSNC M	If (M) =\= 0, skip next instruction	1 or 2	None	1101 1011 MMMM MMMM
TMSNS	If (acc) =\=0, skip next instruction	1 or 2	None	1101 1010 XXXX XXXX
TMCOMPE M	If (acc) =(M), skip next instruction	1 or 2	None	1010 0000 MMMM MMMM
TMCOMPEB M	If (acc) =\=(M), skip next instruction	1 or 2	None	1101 1101 MMMM MMMM

8. 电气特征

8.1 绝对最大额定值

供电电压 …… Vss-0.3V 到 Vss+5.5V　　存储温度 …… –50℃到 125℃

输入电压 …… Vss-0.3V 到 VDD+0.3V　　工作温度 ……–20℃到 70℃

< 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

符号	参数	测试条件		Min.	Typ.	Max.	单位
		VDD	条件				
VDD	工作电压	---		2.2		3.6	V
$I_{DD1.}$	工作电流（32K 晶振）	3V	睡眠模式，看门狗关闭			1	uA
$I_{DD2.}$	工作电流（32K 晶振）	3V	睡眠模式，看门狗开启		2.5		uA
$I_{DD3.}$	工作电流（32K 晶振）	3V	暂停模式，LCD 关闭，看门狗关闭		2.5		uA
$I_{DD4.}$	工作电流（32K 晶振）	3V	暂停模式，LCD 关闭，看门狗开启		2.5		uA
$I_{DD5.}$	工作电流（32K 晶振）	3V	暂停模式，LCD 开启，看门狗关闭		2.5		uA
$I_{DD6.}$	工作电流（32K 晶振）	3V	暂停模式，LCD 开启，看门狗开启		2.5		uA
$V_{IH1.}$	Port PA0~6	3V	输出低电压到高电压		1.7		V
$V_{IL1.}$	Port PA0~6	3V	输入高电压到低电压		1.2		V
$V_{IH2.}$	RESETB	3V	输出低电压到高电压		1.5		V
$V_{IL2.}$	RESETB	3V	输入高电压到低电压		1		V
$V_{IH3.}$	Port PC0~5，CAP1 和 CAP2	3V	输出低电压到高电压		2.2		V
$V_{IL3.}$	Port PC0~5，CAP1 和 CAP2	3V	输入高电压到低电压		1.0		V
$V_{IL4.}$	CAPT1A/CAPT1B 输入低电压	3V	输出低电压到高电压		1		V
$V_{IH4.}$	PA0~6 输入高电压	3.3V	输入高电压到低电压		2.2		V

$I_{.IL.}$	输入泄漏电流	3V	Vin=VDD，VSS			1	μA
$R_{.PH1.}$	上拉电阻	3V	Port PA0~6		95		Kohm
$R_{.PL1.}$	下拉电阻	3V	Port PA0~6，PC0~4，PD0~7		100		Kohm
$R_{.PL2.}$	下拉电阻	3V	RESETB		150		Kohm
$I_{.LVR1.}$	LVR 电流	3V	低电压检测器 配置 bit7.bit6=10		0.3		uA
		5V			1		
$I_{.LVR2.}$	LVR 电流	5V	低电压检测器 配置 bit7.bit6=00		3		uA
$I_{.LVD1.}$	LVD 电流	3V	低电压检测器 配置 bit7.bit6=10		0.3		uA
		5V			1		
$I_{.LVD2.}$	LVD 电流	3V	低电压检测器 配置 bit7.bit6=00		3		uA
$V_{.LVR1.}$	LVR 电压	3V	低电压检测器 配置 bit7.bit6=00		1.5		V
$V_{.LVR2.}$	LVR 电压	3V	低电压检测器 配置 bit7.bit6=01		1.7		V
$V_{.LVR3.}$	LVR 电压	3V	低电压检测器 配置 bit7.bit6=10		2		V
$V_{.LVD1.}$	LVD 电压	3V	SYS_CTL bit4.3=11		2.56		V
$V_{.LVD2.}$	LVD 电压	3V	SYS_CTL bit4.3=10		2.40		V
$V_{.LVD3.}$	LVD 电压	3V	SYS_CTL bit4.3=01		2.68		V
$I_{.OH.}$	PA0~3 输出口， PB0~7 输出口， 驱动电流	5V	Voh=4.5V		9		mA
			Voh=4.0V		18		mA
			Voh=3.5V		22		mA
			Voh=3.0V		26		mA
			Voh=2.5V		29		mA
		3V	Voh=2.7V		4		mA
			Voh=2.4V		7		mA
			Voh=2.1V		9		mA

			Voh=2.1V		10		mA
			Voh=1.8V		12		mA
$I_{OL.}$	PA0~3 输出口， PB0~7 输出口， 流进电流	5V	Voh=0.5V		20		mA
			Voh=1.0V		37		mA
			Voh=1.5V		48		mA
			Voh=2.0V		55		mA
			Voh=2.5V		59		mA
		3V	Voh=0.3V		8		mA
			Voh=0.6V		16		mA
			Voh=0.9V		21		mA
			Voh=1.2V		25		mA
			Voh=1.5V		26		mA

8.3 交流电特性

符号	参数	测试条件		Min	Typ	Max	单位
		条件	VDD				
$f_{.sys1.}$	系统时钟	LP 晶振模式	3.3V		32		Khz
$f_{.sys2.}$	系统时钟	NT 晶振模式	3.3V	0.455		10	Mhz
$f_{.sys3.}$	系统时钟	内部低速 RC	3.3V		50		Khz
$f_{.sys4.}$	系统时钟	内部 700K RC	3.3V		700		Mhz
$f_{.sys5.}$	系统时钟	内部 1.5M RC	3.3V		1.5		Mhz
$f_{.sys5.}$	系统时钟	外部 RC	3.3V			6	Mhz
$T_{.wdt.}$	1 : 1 看门狗定时器	SOSC1.0=00	3.3V	晶体 x 512			mS
		SOSC1.0=01			24		
		SOSC1.0=10		外部 RC x 512			
		SOSC1.0=11			24		
$T_{.rht.}$	复位保留时间		3.3V		24		mS

8.4 外部 RC 表格

低速外部 RC 表格

R 数值	C 数值	RC 频率	R 连接到（VDD，XIN1）
196K	0.1u（建议）	80 Khz	电容器用于稳定频率
295K	0.1u（建议）	54 Khz	
384K	0.1u（建议）	42.56 Khz	
500K	0.1u（建议）	33.3 Khz	
670K	0.1u（建议）	25.3 Khz	

高速外部 RC 表格

R 数值	C 数值	RC 频率	R 连接到（VDD，OSCIN）
383K	0.1u（建议）	203 Khz	电容器用于稳定频率
196K	0.1u（建议）	418 Khz	
147K	0.1u（建议）	570 Khz	
98K	0.1u（建议）	0.90 Mhz	
50.4K	0.1u（建议）	1.89 Mhz	
38.4K	0.1u（建议）	2.58 Mhz	
29.5K	0.1u（建议）	3.46 Mhz	
19.7K	0.1u（建议）	5.46 Mhz	