功能概述

- **高性能低功耗 8 位 LGT8XM 内核**
- **高级 RISC 构架**
  131 条指令，80%以上为单周期执行
  32x8 通用工作寄存器
  32MHz 工作时最高可达 32MIPS 的执行效率
  内部单周期乘法器(8x8)
- **非易失程序与数据存储空间**
  32Kbytes 片上可在线编程 **FLASH** 程序存储器
  2Kbytes 内部数据 **SRAM**
  可编程 E2PROM 模拟接口，支持字节访问
  全新的程序加密算法，保证用户代码安全
- **外设控制器**
  两个具有独立预分频器的 8 位定时器，支持比较输出模式
  两个具有独立预分频器的 16 位定时器，支持输入俘获和比较输出
  内部 32KHz 可校准 **RC** 振荡器实现实时计数器功能
  最多可支持 9 路 PWM 输出，三组互补可编程死区控制
  12 通道 12 位高速模数转换器(**ADC**)
  - 可选内部、外部参考电压
  - 可编程增益(X1/8/16/32)差分放大输入通道
  - 自动阀值电压监控模式

  两路模拟比较器(**AC**)，支持来自 ADC 输入通道的扩展
  内部 1.024V/2.048V/4.096V ±1%可校准参考电压源
  一个 8 位可编程 **DAC**，可用于产生参考电压源
  可编程看门狗定时器（**WDT**）
  可编程同步/异步串行接口（**USART/SPI**）
  同步外设接口(**SPI**)，可编程主/从工作模式
  双线串行接口(**TWI**)，兼容 **I2C** 主从模式
  16 位数字运算加速单元(**DSC**)，支持直接 16 位数据存取访问
- **特殊处理器功能**
  SWD 双线片上调试/量产接口
  外部中断源与 I/O 电平变化中断支持
  内置上电复位电路（**POR**）与可编程低电压检测电路（**LVD**）
  内置 1%可校准 32MHz RC 振荡器，支持倍频输出
  内置 1%可校准 32KHz RC 振荡器
  外部支持 32.768KHz 以及 400K~32MHz 晶振输入
  6x 大电流推挽驱动 IO，支持高速 PWM 应用

LogicGreen

8-bit LGT8XM

RISC Microcontroller with
In-System Programmable
FLASH Memory

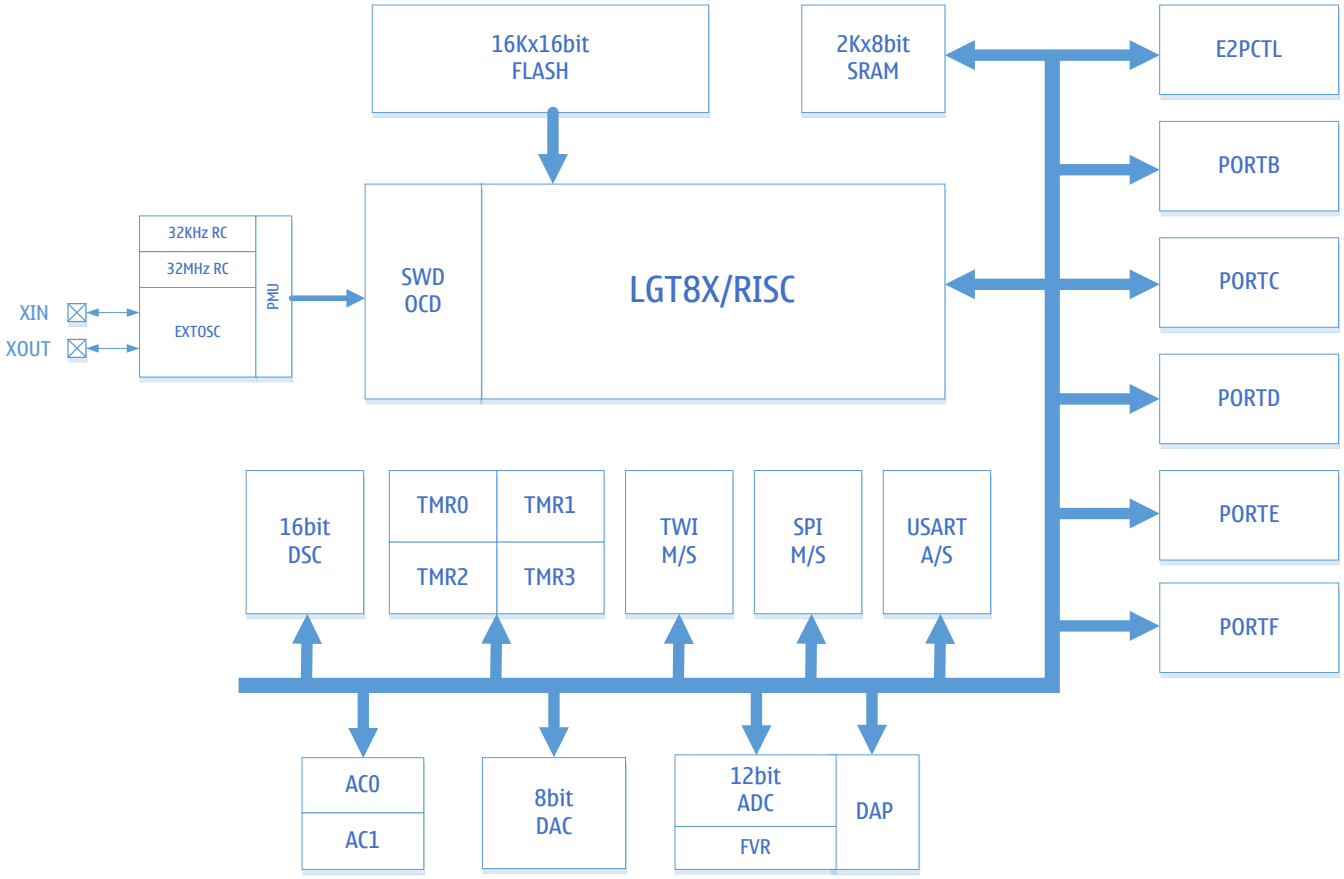
LGT8F88P
LGT8F168P
LGT8F328P

Data book
Version 1.0.3

应用领域
家电
马达驱动
自动化控制

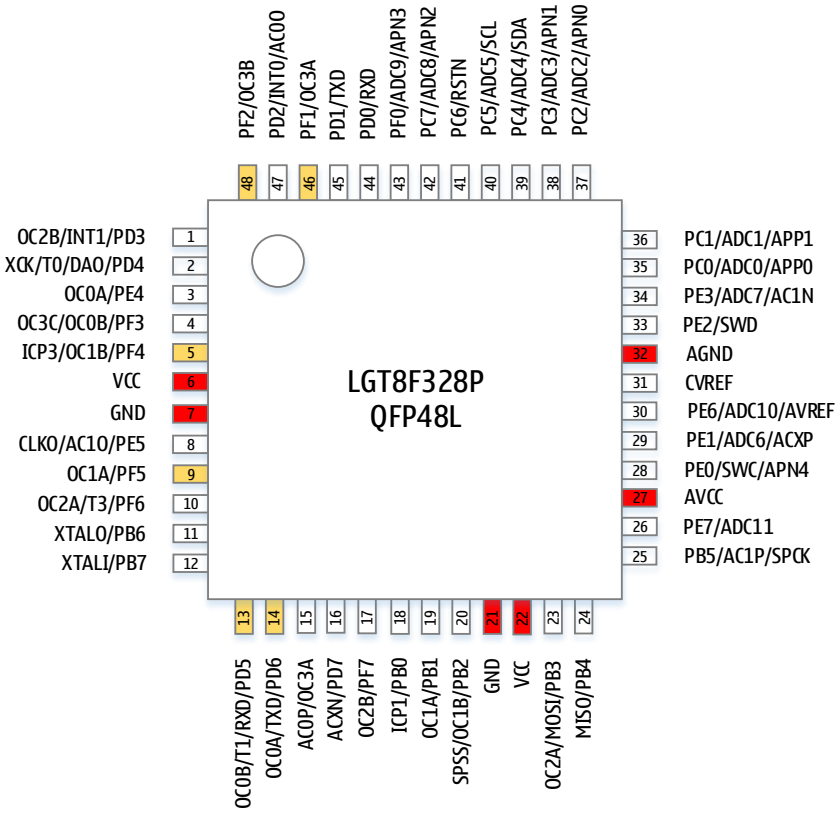
- **I/O 与封装：** QFP48/32L, SSOP20L
- **最低功耗：** 1uA@3.3V
- **工作环境**
  工作电压： 1.8V ~ 5.5V
  工作频率： 0 ~ 32MHz
  工作温度： -40C ~ +85C
  HBM ESD： > 4KV

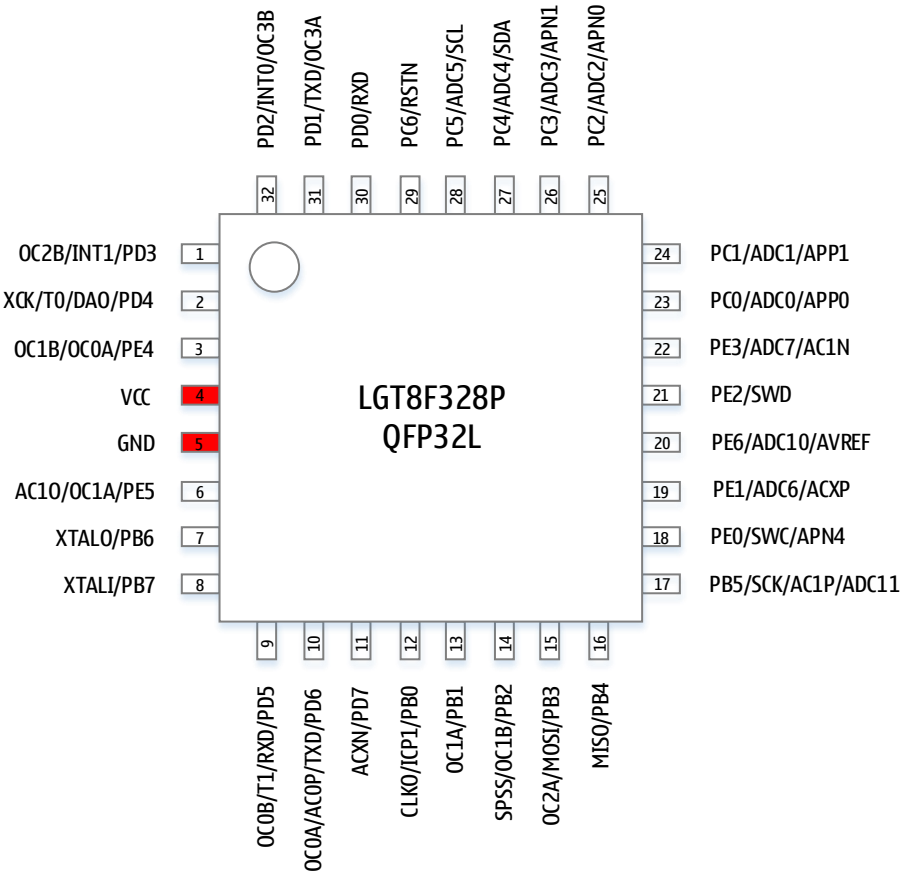
系统框架

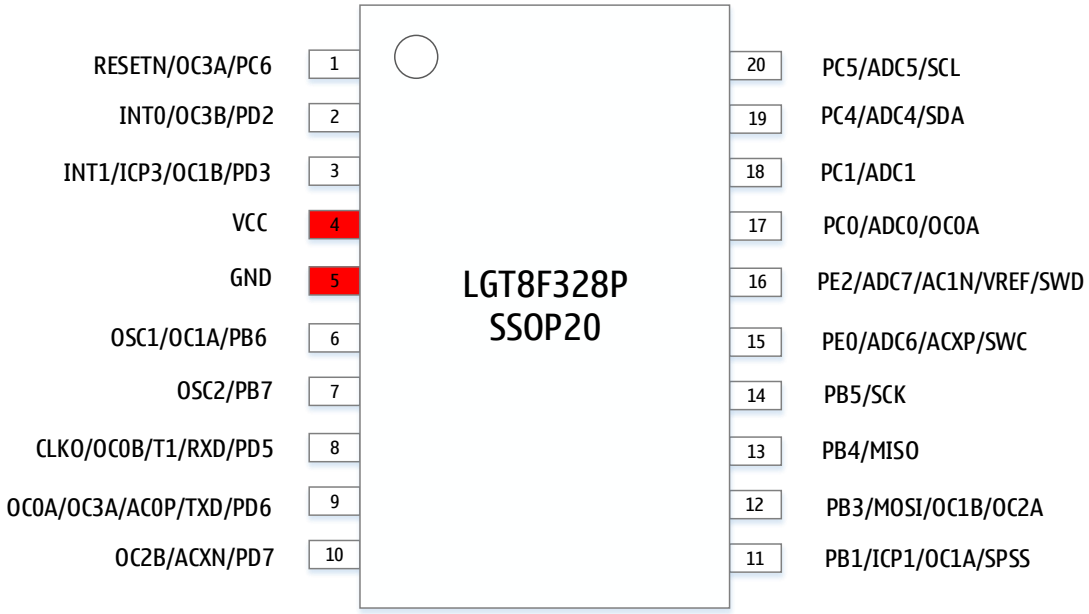


模块名称	模块功能
SWD	调试模块，同时实现在线调试与 ISP 功能
LGT8X	8bit 高性能 RISC 内核
E2PCTL	数据 FLASH 访问接口控制器
PMU	功耗管理模块， 负责管理系统工作状态之间的转换
PORTB/C/D/E/F	通用可编程输入输出端口
DSC	16 位数字运算加速单元
ADC DAP IVREF	8 通道 12 位模数转换器 可编程增益差分放大器 1.024V/2.048V/4.096V 内部参考
AC0/1	模拟比较器
TMR0/1/2/3	8/16 位定时/计数器，PWM 控制器
WDT	看门狗复位模块
SPI M/S	主从 SPI 控制器
TWI M/S	主从双线接口控制器，兼容 I2C 协议
USART	同步/异步串行收发器
DAC	8 位数模转换器

封装定义


PF2/OC3B 48
PD2/INT0/AC0O 47
PF1/OC3A 46
PD1/TXD 45
PD0/RXD 44
PF0/ADC9/APN3 43
PC7/ADC8/APN2 42
PC6/RSTN 41
PC5/ADC5/SCL 40
PC4/ADC4/SDA 39
PC3/ADC3/APN1 38
PC2/ADC2/APN0 37
OC2B/INT1/PD3 1
XCK/T0/DAO/PD4 2
OC0A/PE4 3
OC3C/OC0B/PF3 4
ICP3/OC1B/PF4 5
VCC 6
GND 7
CLKO/AC1O/PE5 8
OC1A/PF5 9
OC2A/T3/PF6 10
XTALO/PB6 11
XTALI/PB7 12
LGT8F328P
QFP48L
36 PC1/ADC1/APP1
35 PC0/ADC0/APP0
34 PE3/ADC7/AC1N
33 PE2/SWD
32 AGND
31 CVREF
30 PE6/ADC10/AVREF
29 PE1/ADC6/ACXP
28 PE0/SWC/APN4
27 AVCC
26 PE7/ADC11
25 PB5/AC1P/SPCK
13 OC0B/T1/RXD/PD5
14 OC0A/TXD/PD6
15 AC0P/OC3A
16 ACXN/PD7
17 OC2B/PF7
18 ICP1/PB0
19 OC1A/PB1
20 SPSS/OC1B/PB2
21 GND
22 VCC
23 OC2A/MOSI/PB3
24 MISO/PB4


PD2/INT0/OC3B 32
PD1/TXD/OC3A 31
PD0/RXD 30
PC6/RSTN 29
PC5/ADC5/SCL 28
PC4/ADC4/SDA 27
PC3/ADC3/APN1 26
PC2/ADC2/APN0 25
OC2B/INT1/PD3 1
XCK/T0/DAO/PD4 2
OC1B/OC0A/PE4 3
VCC 4
GND 5
AC1O/OC1A/PE5 6
XTALO/PB6 7
XTALI/PB7 8
LGT8F328P
QFP32L
24 PC1/ADC1/APP1
23 PC0/ADC0/APP0
22 PE3/ADC7/AC1N
21 PE2/SWD
20 PE6/ADC10/AVREF
19 PE1/ADC6/ACXP
18 PE0/SWC/APN4
17 PB5/SCK/AC1P/ADC11
9 OC0B/T1/RXD/PD5
10 OC0A/AC0P/TXD/PD6
11 ACXN/PD7
12 CLKO/ICP1/PB0
13 OC1A/PB1
14 SPSS/OC1B/PB2
15 OC2A/MOSI/PB3
16 MISO/PB4


RESETN/OC3A/PC6
1
INT0/OC3B/PD2
2
INT1/ICP3/OC1B/PD3
3
VCC
4
GND
5
OSC1/OC1A/PB6
6
OSC2/PB7
7
CLKO/OC0B/T1/RXD/PD5
8
OC0A/OC3A/AC0P/TXD/PD6
9
OC2B/ACXN/PD7
10
LGT8F328P
SSOP20
20
PC5/ADC5/SCL
19
PC4/ADC4/SDA
18
PC1/ADC1
17
PC0/ADC0/OC0A
16
PE2/ADC7/AC1N/VREF/SWD
15
PE0/ADC6/ACXP/SWC
14
PB5/SCK
13
PB4/MISO
12
PB3/MOSI/OC1B/OC2A
11
PB1/ICP1/OC1A/SPSS

引脚说明

LGT8FX8P 系列封装中，QFP48L 封装引出全部引脚。其他封装均为在 QFP48 基础上将多个内部 I/O 绑定到一个引脚上产生。配置引脚方向时需特别注意。下表列出各种封装引脚的绑定情况：

QFP48	QFP32	SSOP20	功能说明
01	01	03	PD3/INT1/OC2B*
			PD3: 可编程端口 D3 INT1: 外部中断输入 1 OC2B: 定时器 2 比较匹配输出 B
02	02		PD4/DAO/T0/XCK
			PD4: 可编程端口 D4 DAO: 内部 DAC 输出 T0: Timer0 外部时钟输入 XCK: USART 同步传输时钟
03	03	-	PE4/OC0A*
			PE4: 可编程端口 E4 OC0A: 定时器 0 比较匹配输出 A
04	-	-	PF3/OC3C/OC0B*
			PF3: 可编程端口 F3 OC3C: 定时器 3 比较匹配输出 C OC0B: 定时器 0 比较匹配输出 B
05	03	03	PF4/OC1B*/ICP3
			PF4: 可编程端口 F4 OC1B: 定时器 1 比较匹配输出 B ICP3: 定时器 3 俘获输入
06	04	04	VCC
07	05	05	GND
08	06	-	PE5/AC1O/CLKO*
			PE5: 可编程端口 E5 C1O: 模拟比较器 AC1 输出 CLKO: 系统时钟输出
09		06	PF5/OC1A*
			PF5: 可编程端口 F5 OC1A: 定时器 1 比较匹配输出 A
10	-	-	PF6/T3/OC2A*
			PF6: 可编程端口 F6 T3: 定时器 3 外部时钟输入 OC2A: 定时器 2 比较匹配输出 A
11	07	06	PB6/XTALO
			PB6: 可编程端口 B6 XTALO: 晶振 IO 输出端口

12	08	07	PB7/XTALI PB7: 可编程端口 B7` XTALI: 晶振 IO 输入端口
13	09	08	PD5/RXD*/T1/OC0B PD5: 可编程端口 D5 RXD: USART 数据接收(备选) T1: 定时器 1 外部时钟输入 OC0B: 定时器 0 比较匹配输出 B
14	10	09	PD6/TXD*/OC0A PD6: 可编程端口 D6 TXD: USART 数据发送(备选) OC0A: 定时器 0 比较匹配输出 A
15			AC0P/OC3A AC0P: 模拟比较器 0 正端输入 OC3A: 定时器 3 比较匹配输出 A
16	11	10	PD7/ACXN PD7: 可编程端口 D7 ACXN: 模拟比较器 0/1 公用负端输入
17	-		PF7/OC2B PF7: 可编程端口 F7 OC2B: 定时器 2 比较匹配输出 B
18	12	11	PB0/ICP1 PB0: 可编程端口 B0 ICP1: 定时器 1 俘获输入
19	13		PB1/OC1A PB1: 可编程端口 B1 OC1A: 定时器 1 比较匹配输出 A
20	14	12	PB2/OC1B/SPSS PB2: 可编程端口 B2 OC1B: 定时器 1 比较匹配输出 B SPSS: SPI 从机模式片选
21	-	-	GND
22	-	-	VCC
23	15	12	PB3/MOSI/OC2A PB3: 可编程端口 B3 MOSI: SPI 主机输出/从机输入 OC2A: 定时器 2 比较匹配输出 A
24	16	13	PB4/MISO PB4: 可编程端口 B4 MISO: SPI 主机输入/从机输出
25	17	14	PB5/SPCK/AC1P PB5: 可编程端口 B5 SPCK: SPI 时钟信号 AC1P: 模拟比较器 1 正端输入

26	-	-	PE7/ADC11 PE7: 可编程端口 E7 ADC11: ADC 模拟输入通道 11
27	-	-	AVCC: 内部模拟电路电源
28	18	15	PE0/SWC/APN4 PE0: 可编程端口 E0 SWC: SWD 调试接口时钟 APN4: 差分放大器反向输入通道 4
29	19		PE1/ADC6/ACXP PE1: 可编程端口 E1 ADC6: ADC 模拟输入通道 6 ACXP: 模拟比较器 0/1 公用正端输入
30	20	16	PE6/ADC10/AVREF PE6: 可编程端口 E6 ADC10: ADC 模拟输入通道 10 AVREF: ADC 外部参考输入
31	-	-	CVREF: ADC 参考电压输出 只用于外接 0.1uF 滤波电容
32	-	-	AGND: 内部模拟电路地
33	21	16	PE2/SWD PE2: 可编程端口 E2 SWD: SWD 调试接口数据线
34	22		PE3/ADC7/AC1N PE3: 可编程端口 E3 ADC7: ADC 模拟输入通道 7 AC1N: 模拟比较器负端输入
35	23	17	PC0/ADC0/APP0 PC0: 可编程端口 C0 ADC0: ADC 模拟输入通道 0 APP0: 差分放大器正向输入通道 0
36	24	18	PC1/ADC1/APP1 PC1: 可编程端口 C1 ADC1: ADC 模拟输入通道 1 APP1: 差分放大器正向输入通道 1
37	25	-	PC2/ADC2/APN0 PC2: 可编程端口 C2 ADC2: ADC 模拟输入通道 2 APN0: 差分放大器反向输入通道 0
38	26	-	PC3/ADC3/APN1 PC3: 可编程端口 C3 ADC3: ADC 模拟输入通道 3 APN1: 差分放大器反向输入通道 1

39	27	19	PC4/ADC4/SDA PC4: 可编程端口 C4 ADC4: ADC 模拟输入通道 4 SDA: I2C 控制器数据线
40	28	20	PC5/ADC5/SCL PC5: 可编程端口 C5 ADC5: ADC 模拟输入通道 5 SCL: I2C 控制器时钟线
41	29	1	PC6/RESETN PC6: 可编程端口 C6 RESETN: 外部复位输入
42	-	-	PC7/ADC8/APN2 PC7: 可编程端口 C7 ADC8: ADC 模拟输入通道 8 APN2: 差分放大器反向输入通道 2
43	-	-	PF0/ADC9/APN3 PF0: 可编程端口 F0 ADC9: ADC 模拟输入通道 9 APN3: 差分放大器反向输入通道 3
44	30	-	PD0/RXD PD0: 可编程端口 D0 RXD: USART 数据接收输入
45	31	-	PD1/TXD PD1: 可编程端口 D1 TXD: USART 数据发送输出
46		1	PF1/OC3A PF1: 可编程端口 F1 OC3A: 定时器 3 比较匹配输出 A
47	32	2	PD2/INT0/AC0O PD2: 可编程端口 D2 INT0: 外部中断输入 0 AC0O: 模拟比较 0 输出
48			PF2/OC3B PF2: 可编程端口 F2 OC3B: 定时器 3 比较匹配输出 B

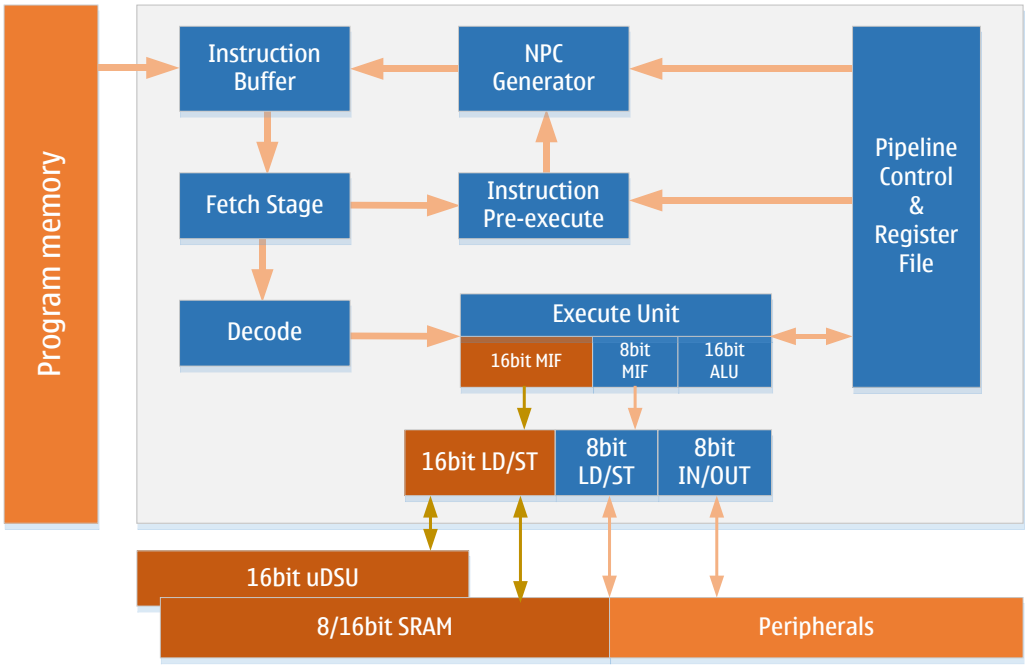
LGT8XM 内核

- 低功耗设计
- 高效率 RISC 构架
- 16 位 LD/ST 扩展(uDSU 专用)
- 130 条指令，其中 80%以上为单周期
- 内嵌在线调试(OCD)支持

概述

本章节主要描述 LGT8XM 内核构架和功能。 内核是 MCU 的大脑，负责保证程序的正确执行，因此内核必须能够准确的执行计算，控制外设以及处理各种中断。

LGT8XM 内核的结构



为了实现更大的效率和并行性，LGT8XM 内核采用哈弗构架 – 独立的数据和程序总线。指令通过一个优化的两级流水线执行，两级流水线能够减少流水线中无效指令的个数，减少了对 FLASH 程序存储器的访问量，因此可以降低内核运行的功耗。同时 LGT8XM 内核在取指令的前级中增加了指令缓存（可以同时缓存 2 条指令），通过在取指令周期的预执行模块，进一步减少了对 FLASH 程序存储器的访问频率；经大量测试，LGT8XM 可以比其他同类构架的内核减少约 50%对 FLASH 的访问，大大降低了系统的运行功耗。

LGT8XM 内核具有 32 个 8 位高速访问的通用工作寄存器(Register file)，有助于实现单周期的算术逻辑运算(ALU)。一般情况下，ALU 运算的两个操作数均来自与通用工作寄存器，ALU 运算的结果也会在一个周期内写入到寄存器文件中。

32 个通过工作寄存器中的 6 个用于两两结合构成三个 16 位寄存器，可用于间接寻址地址指针，用于访问外部存储空间以及 FLASH 程序空间。LGT8XM 支持单周期的 16 位算术运算，极大的提高了间接寻址的效率。LGT8XM 内核中这三个特殊的 16 位寄存器被命名为 X, Y, Z 寄存器，将在后面详细介绍。

ALU 支持寄存器之间以及常数与寄存器之间的算术逻辑运算，单个寄存器的运算也可以在 ALU 中执行。ALU 运算完成后，运算结果对内核状态的影响更新到状态寄存器中(SREG)。

程序流程控制通过条件和无条件跳转/调用实现，可以寻址到所以的程序区域。大部分 LGT8XM 指令为 16 位。每个程序地址空间对应一个 16 位或者 32 位的 LGT8XM 指令。

内核响应中断或子程序调用后，返回地址(PC)被存储在堆栈中。堆栈被分配在系统的一般数据 SRAM 中，因此堆栈的大小仅受限于系统中 SRAM 的大小和用法。所有的支持中断或子程序调用的应用，必须首先初始化堆栈指针寄存器(SP)，SP 可以通过 IO 空间访问。数据 SRAM 可以通过 5 种不同的寻址模式访问。LGT8XM 的内部存储空间都被线性的映射到一个统一的地址空间。具体请参考存储章节的介绍。

LGT8XM 内核包含了一个灵活的中断控制器，中断功能可以通过状态寄存器中的一个全局中断使能位控制。所有的中断都有一个独立的中断向量。中断的优先级与中断向量地址有对应关系，中断地址越小，中断的优先级就越高。

I/O 空间包含了 64 个可以通过 IN/OUT 指令直接寻址的寄存器空间。这些寄存器现实对内核控制以及状态寄存器，SPI 以及其他 I/O 外设的控制功能。这部分空间可以通过 IN/OUT 指令直接访问，也可以通过他们映射到数据存储器空间的地址访问(0x20 – 0x5F)。另外，LGT8FX8P 也包含扩展的 I/O 空间，他们被映射到数据存储空间 0x60 – 0xFF，这里只能使用 ST/STS/STD 以及 LD/LDS/LDD 指令访问。

为增强 LGT8XM 内核的运算能力 ，指令流行线中增加了 16 位的 LD/ST 扩展。此 16 位 LD/ST 扩展配合 16 数字运算加速单元(uDSU)工作，实现高效的 16 位数据运算。同时内核也增加对 RAM 空间的 16 位访问能力。因此 16 位 LD/ST 扩展可以在 uDSU, RAM，以及工作寄存器之间传递 16 位的数据。具体细节请参考”数字运算加速器”章节。

算术逻辑运算单元（ALU）

LGT8XM 内部包含了一个 16 位的算术逻辑运算单元，能够在一个周期内完成 16 为数据的算术运算。高效的 ALU 与 32 个通用工作寄存器相连。能够在一个周期内完成两个寄存器或者寄存器与立即数之间的算术逻辑运算。ALU 的运算分为三种：算术，逻辑以及位运算。同时 ALU 部分也包含了一个单周期的硬件乘法器，能够在一个周期内实现两个 8 位寄存器直接的有符号或者无符号运算。请参考指令集部分的详细介绍。

状态寄存器（SREG)

状态寄存器中主要保存了因执行最近一次 ALU 运算而产生的结果信息。这些信息用于控制程序执行流程。状态寄存器是在 ALU 操作完全结束后更新，这样就可以省去了使用单独的比较指令，可以带来更加紧凑高效的代码实现。状态寄存器的值在响应中断和从中断中退出时并不会自动保存和恢复，这需要软件去实现。

SREG 寄存器定义

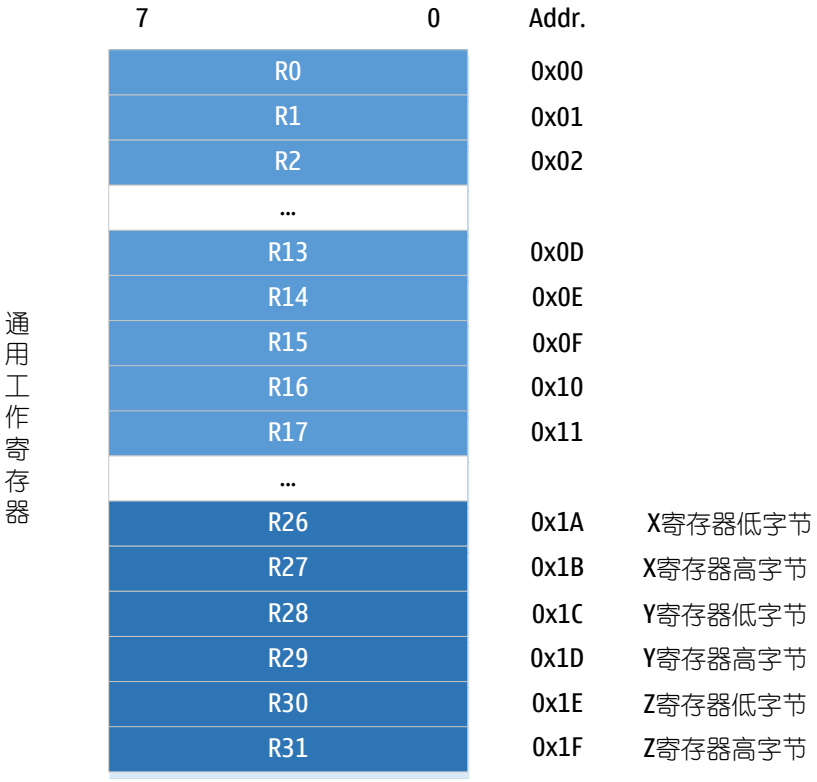
SREG 系统状态寄存器								
地址: 0x3F (0x5F)				默认值: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	I	T	H	S	V	N	Z	C
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[0]	C	进位标志，表示算术或逻辑操作导致了进位，具体请参考指令描述						
[1]	Z	零标志，表示算术或逻辑运算的结果为零，请参考指令描述部分						
[2]	N	负标志，表示算术或逻辑运算产生了一个负数，请参考指令描述部分						
[3]	V	溢出标志，表示二进制补码运算结果产生溢出，请参考指令描述部分						
[4]	S	符号位，等效于 N 与 V 的异或运算结果，具体请参考指令描述部分						
[5]	H	半进位标志，在 BCD 运算中有用，表示字节运算产生了的半进位						
[6]	T	临时位，位复制(BLD)和位存储(BST)指令中使用，T 位将作为一个临时的存储位，用于临时存放通用寄存器中的某一位的值。具体请参考指令描述部分						
[7]	I	全局中断使能位，必须设置此位为 1 才能使能内核响应中断事件。不同的中断源是由独立的控制位控制。全局中断使能位是控制中断信号进入内核的最后一道屏障。I 位在内核响应中断向量后由硬件自动清除，在执行中断返回指令(RETI)后自动置位。I 位也可以使用 SEI 和 CLI 指令改变，请参考指令描述部分						

通用工作寄存器

通用工作寄存器根据 LGT8XM 指令集构架优化。为了达到内核执行需要的效率和灵活性，LGT8XM 内部的通用工作寄存器支持一下几种访问模式：

- 一个 8 位的读同时一个 8 位的写操作
- 两个 8 位的读同时一个 8 位的写操作
- 两个 8 位的读同时一个 16 位的写操作
- 一个 16 位的读同时一个 16 位的写操作

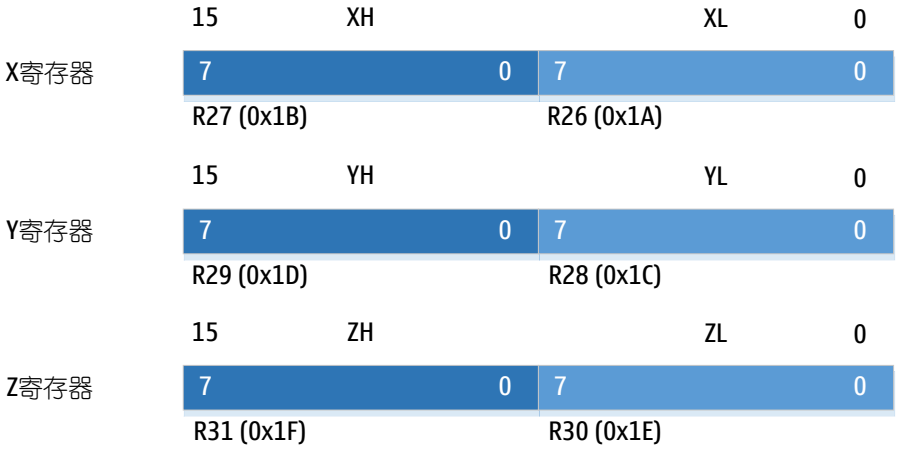
LGT8XM 通用工作寄存器



大部分指令能够直接访问到全部的通用工作寄存器，他们大部分也都是单周期指令。
如上图所示，每一个寄存器都对应一个数据存储空间的地址，这些通用工作寄存器被映射到数据储存空间。尽快他们不没有真正的存在于 SRAM 中，但这种统一映射的存储组织给访问他们带来了很大的灵活性。X/Y/Z 寄存器可以作为指针索引到任何通用寄存器。

X/Y/Z 寄存器

寄存器 R26...R31 可以两两组合，构成三个 16 位寄存器。这三个 16 位寄存器主要用于间接寻址访问的地址指针，X/Y/Z 寄存器结构如下：



在不同的寻址模式下， 这些寄存器被用作固定偏移，自动递增以及自动递减的地址指针，具体细节请参考指令描述部分。

堆栈指针

堆栈用于存储临时数据，局部变量以及中断和子程序调用的返回地址。需要特别注意的是，堆栈别设计为从高地址向低地址生长。堆栈指针寄存器(SP)总是指向堆栈的顶部。堆栈指针指向数据 SRAM 所在的物理空间，这里存放子程序或中断调用必须的堆栈空间。PUSH 指令将会使得堆栈指针递减。

堆栈在 SRAM 中的位置必须在子程序执行或者中断使能之前由软件正确的设置。一般情况下是将堆栈指针初始化指向 SRAM 的最高地址处。堆栈指针必须设置为高位 SRAM 开始地址。SRAM 在系统数据存储映射的地址请参考系统数据存储部分。

堆栈指针相关的指令

指令	堆栈指针	描述
PUSH	增加 1	数据压入堆栈
CALL ICALL RCALL	增加 2	中断或者子程序调用的返回地址压入堆栈
POP	减少 1	数据从堆栈取出
RET RETI	减少 2	中断或者子程序调用的返回地址从堆栈中取出

堆栈指针由分配在 I/O 空间的两个 8 位的寄存器构成。堆栈指针的实际长度与系统实现相关。在 LGT8XM 构架的有些芯片实现中，数据空间非常小，以至于仅仅 SPL 就能满足寻址需要，这种情况下，SPH 寄存器将不会出现。

SPH/SPL 堆栈指针寄存器定义

SPH/SPL 堆栈指针寄存器		
SPH: 0x3E (0x5E) SPL: 0x3D (0x5D)		默认值: RAMEND
SP	SP[15:0]	
R/W	R/W	
位定义		
[7:0]	SPL	堆栈指针低 8 位
[15:8]	SPH	堆栈指针高 8 位

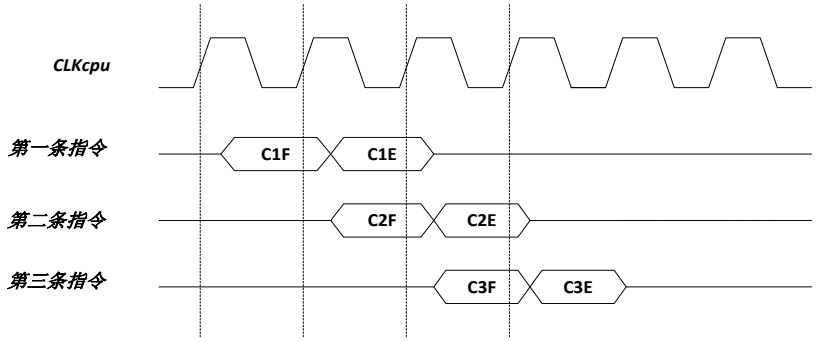
指令执行时序

这一章节描述指令执行的一般时序概念。LGT8XM 内核由内核时钟(CLKcpu)驱动，这个时钟直接来自与系统的时钟源选择电路。

下图展示了哈弗构架与快速访问寄存器文件概念基础上的指令流水线执行时序。这是使

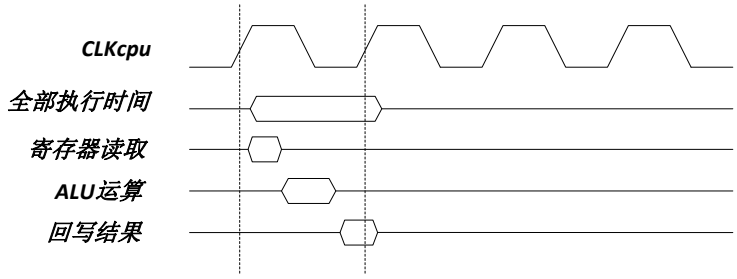
得内核能够获得 1MIPS/MHz 的执行效率的物理保证。

从上图可以看出，第一条指令的执行期间同时会读出第二条指令。当第二条指令进入执



行期间，同时又会读出第三条指令。这样在整个执行期间，并不需要为读取指令花费额外的周期，从流水线上看，实现了每个周一执行一条指令的效率。

下图展示通用工作寄存器的访问时序，在一个周期内，ALU 操作使用到两个寄存器作为操作数，并在这个周期内将 ALU 执行结果写入到目标寄存器中。



复位与中断处理

LGT8XM 支持多个中断源。这些中断以及复位向量在程序空间都对应一个独立的程序向量入口。一般而言，所有的中断都有单独的控制位控制。当设置了该控制位，并且使能了内核的全局中断使能位后，内核才能响应这个中断。

最低的程序空间默认保留为复位以及中断向量区域。LGT8FX8P 支持的完整的中断列表请参考中断章节的介绍。这个列表同时也决定了不同中断的优先级。向量地址越低的中断，对应的中断优先级就越高。复位(RESET)具有最高的优先级，然后是 INT0 – 外部中断请求 0. 中断向量表的起始地址（复位向量除外）可以被重新定义到任何 256 字节对齐的开始处，需要通过 MCU 控制寄存器(MCUCR)中的 IVSEL 位以及 IVBASE 向量基地址寄存器实现。

当内核响应中断后，全局中断使能标志为 I 会被硬件自动清除。用户可以通过将 I 位使能实现中断嵌套。这样任何随后发生的中断都会中断当前的中断服务程序。I 位在执行中断返回指令(RETI)后自动置位，从而可以正常响应随后发生的中断。

有种基本的中断类型。第一种类型由事件触发，中断事件发生后置位中断标志位。对于这种中断来说，内核响应中断请求后，当前的 PC 值被直接替换为实际的中断向量地址，执行对应的中断服务子程序，同时硬件自动清除掉中断标志位。中断标志位也可以通过向中断标志位的位置写 1 清除。如果在发生中断时，中断使能位被清除，中断标志位仍然会被设置以记录中断事件。等到中断使能后，这个记录的中断事件会被立即响应。同样，如果在中断发生时，全局中断使能位(SERG.I)被清除，对应的中断标志位也会被设置以记录中断事件，等

到全局中断使能位被设置后，这些被记录的中断将会依照优先级依次执行。

第二种中断类型是当中断条件一直存在时，中断就一直响应。这种中断不需要中断标志位。如果中断条件在中断使能之前消失，这个中断将不会得到响应。

当 LGT8XM 内核从中断服务子程序中退出后，执行流程会返回到主程序中。在主程序中执行一条或几条指令后，才能响应其他等待的中断请求。

需要注意的是，系统状态寄存器(SREG)在进入中断服务后并不会自动保存，也不会在从中断服务返回后自动恢复。它必须由软件负责处理。

当使用 CLI 指令禁止中断后，中断将会被立即禁止。在 CLI 指令之后发生的所以中断都不会得到响应。即使是和 CLI 指令执行时同时发生的中断，也不会被响应。下面的例子中说明如何利用 CLI 避免中断打乱 EEPROM 的写时序：

中断响应时间

LGT8XM 内核针对中断响应进行了优化，使得任何中断在 4 个系统时钟周期内一定得到响应。4 个系统时钟周期后，中断服务子程序进入执行周期。在这 4 个时钟内，中断之前的 PC 值被压入堆栈，系统执行流程流程跳转到中断向量对应中断服务程序。如果中断发生在一个多周期指令执行期间，内核将保证当前指令正确的执行结束。如果中断发生在系统处于休眠状态下(SLEEP)，中断响应需要额外增加 4 个时钟周期。这增加的时钟周期用于从选择的休眠模式下唤醒操作的同步周期。休眠模式的具体描述，请参考功耗管理的相关章节。

从中断服务子程序中返回需要 2 个时钟周期。在这 2 个时钟周期内，PC 从堆栈中恢复，堆栈指针加 2，并自动使能全局中断控制位。

存储单元

概述

本章节主要描述 LGT8FX8P 系列内部不同的存储单元。LGT8XM 构架支持两种主要的内部存储空间，分别是数据存储空间和程序存储空间。LGT8FX8P 内部也包含了数据 FLASH，通过内部的控制器可以实现 EEPROM 接口的数据存储功能。另外，LGT8FX8P 系统中还包含了特殊的存储单元，用于存放系统配置信息以及芯片的全局设备号(GUID)。

LGT8FX8P 系列芯片包含了 LGT8F88P/168P/328P 四种不同的型号；四种型号的外设以及封装完全兼容，所不同是 FLASH 程序存储空间以及内部数据 SRAM，下面的表格比较清楚的描述了 LGT8FX8P 系列芯片不同的存储空间配置：

DEVICE	FLASH	SRAM	E2PROM	中断向量
LGT8F88P	8KB	1KB	2KB	1 个指令字
LGT8F168P	16KB	1KB	4KB	2 个指令字
LGT8F328P	32KB	2KB	可配置为 0K/1K/2K/4K/8K (与 FLASH 共享)	2 个指令字

LGT8F328P 内部没有独立用于模拟 E2PROM 接口的 FLASH 空间；用于模拟 E2PROM 的存储空间与程序 FLASH 共享，用户可以根据应用需求，选择合适的配置。

由于模拟 E2PROM 接口采用的独特实现，系统需要两倍的程序 FLASH 空间模拟 E2PROM 存储空间，比如对于 LGT8F328P， 如果用户配置了 1KB 的 E2PROM 空间，将会有 2KB 字节的程序空间被保留，剩下 30KB 的 FLASH 空间用于存储程序。

LGT8F328P 程序 FLASH 与 E2PROM 共享配置表：

DEVICE	FLASH	E2PROM
LGT8F328P	32KB	0KB
	30KB	1KB
	28KB	2KB
	24KB	4KB
	16KB	8KB

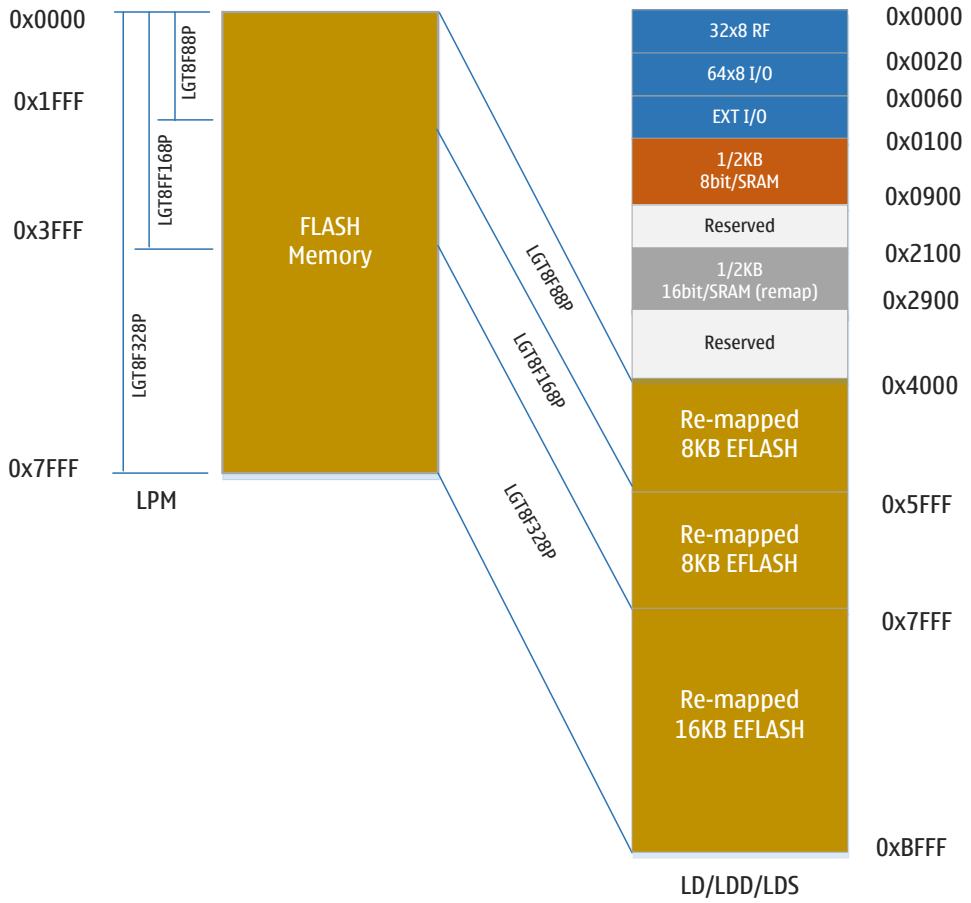
系统可编程 FLASH 程序存储单元

LGT8FX8P 系列微控制器内部分别包括 8K/16K/32K 字节的片上在线可编程 FLASH 程序存储单元。

程序 FLASH 能保证至少 100,000 次以上的擦写周期。LGT8FX8P 内部集成 FLASH 接口控制器，能够实现在系统编程(ISP)以及程序的自升级功能。具体实现细节请参考本章在关于 FLASH 接口控制器部分的描述。

程序空间也可以通过 LPM 指令直接访问(读取)，这个特点可以实现应用相关的常数查找

表。同时 FLASH 程序空间也被映射到系统数据存储空间内，用户也可以使用 LD/LDD/LDS 实现对 FLASH 空间的访问。程序空间被映射到数据存储空间 0x4000 开始的地址范围内。如下图所示：



SRAM 数据存储单元

LGT8FX8P 系列微控制器是一种相对复杂的微控制器，它支持多种不同类型的外设，这些外设的控制器被分配在 64 个 I/O 寄存器空间内。可以直接通过 IN/OUT 指令访问。另一些外设的控制寄存器分配在 0x60 ~ 0xFF 区域内，由于这部分空间是映射到数据存储空间内，只能通过 ST/STS/STD 以及 LD/LDS/LDD 等指令访问。

LGT8FX8P 的系统数据存储空间从 0 地址开始，分别映射了通用工作寄存器文件，I/O 空间，扩展 I/O 空间以及内部数据 SRAM 空间。最开始的 32 个字节地址对应 LGT8XM 内核 32 个通用工作寄存器。接下来的 64 个地址是可以通过 IN/OUT 指令直接访问的标准 I/O 空间。然后的 160 个地址是扩展 I/O 空间，在接下来就是最多 2K 字节的数据 SRAM。从 0x4000 开始到 0xBFFF 结束的这部分空间，映射了 FLASH 程序存储单元。

系统内 1K/2K 字节 SRAM 被分别映射到两个空间。从 0x0100 开始到 0x0900 结束的这个空间被内核以 8 位字节的宽度读写。从 0x2100 开始到 0x2900 结束这个区域为 16 位宽度的访问空间。系统 RAM 被映射到 0x2100 开始的高位地址主要用于配合 uDSU 模块工作，实现高效的 16 位数据存储。在编程时，将普通的 8 位寻址变量地址加上 0x2000 的偏移量，即可切换到 16 位访问模式。

系统支持 5 种不同的寻址模式可以覆盖到整个数据空间：直接访问，带偏移的间接访问，间接访问，访问前递减地址的间接访问，访问后递增地址的间接访问。通用工作寄存器 R26 到 R31 用于间接访问的地址指针。间接访问可以寻址整个数据存储空间。带偏移地址的间接访问能够寻址到以 Y/Z 寄存器为基地址的附近 63 个地址空间。

当使用支持地址自动递增/递减的寄存器间接访问模式，地址寄存器 X/Y/Z 会在访问发生前/后自动由硬件递减/递增。具体请参考指令集描述部分。

16 位寄存器 X/Y/Z 以及与之相关的自动寻址模式(递增、递减)，在 16 位扩展模式下也有着非常重要的作用。16 位扩展模式可以使用 LD/ST 的递增/递减模式，实现带变量的自动递增、递减寻址。这种模式在对数组进行运算操作时，将非常有效。具体实现请参考“数字运算加速器(uDSU)”相关章节。

通用 I/O 寄存器

LGT8FX8P 的 I/O 空间有三个通用 I/O 寄存器 GPIOR2/1/0，这三个寄存器可以使用 IN/OUT 指令访问，用于存放用户自定义数据。

外设寄存器空间

I/O 空间的详细定义，请参考 LGT8FX8P 数据手册中“寄存器概述”章节。

LGT8FX8P 所以的外设都被分配到 I/O 空间。所有的 I/O 空间地址都可以被 LD/LDS/LDDD 以及 ST/STS/STD 指令访问。访问的数据都是通过 32 个通用工作寄存器传递。在 0x00～0x1F 之间的 I/O 寄存器可以通过位寻址指令 SBI 和 CBI 访问。在这些寄存器中，某一个位的值可以使用 SBIS 和 SBIC 指令检测，用以控制程序的执行流程。具体请参考指令集描述部分。

当使用 IN/OUT 指令访问 I/O 寄存器时，必须寻址 0x00 ~ 0x3F 之间的地址。当使用 LD 或 ST 指令访问 I/O 空间时，必须通过 I/O 空间在系统数据存储器统一映射空间的映射地址访问(加上 0x20 的偏移)。其他一些分配在扩展 I/O 空间的外设寄存器(0x60 ~ 0xFF)， 只能够使用 ST/STS/STD 和 LD/LDS/LDD 指令访问。

为了与未来的设备兼容，保留位在写操作时必须写 0。不能在保留的 I/O 空间上执行写操作。

一些寄存器中包括了状态标志，需要被写 1 才能清零。需要注意的是，CBI 和 SBI 指令仅仅支持特定的位，因此 CBI/SBI 也只能工作在包含这些状态标志的寄存器上。除此之外，CBI/SBI 指令只能工作在 0x00 到 0x1F 这个地址范围内的寄存器。

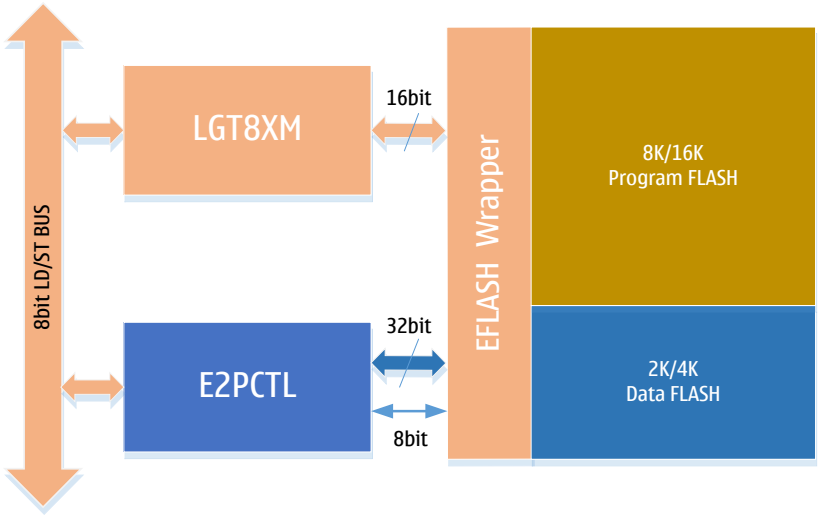
FLASH 控制器(E2PCTL)

LGT8FX8P 内部实现集成了一个灵活可靠的 EFLASH 读写控制器，可以利用系统中已有的数据 FLASH 存储空间，实现字节读写访问的存储空间，实现类似 E2PROM 的存储应用；E2PROM 接口模拟采用擦写均衡的算法，可以将数据 FLASH 的使用周期提高 1 倍左右，能够保证 100,000 次以上的擦写周期。

E2PCTL 控制器也实现了对 FLASH 程序空间的在线擦写操作，可以通过软件实现在线自

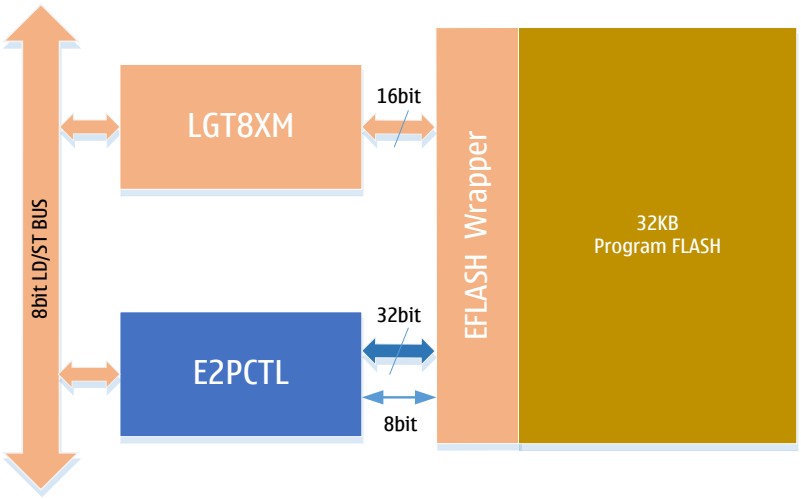
动升级固件的功能。通过 FLASH 控制器访问程序 FLASH 程序空间，只支持页擦除(1024 字节)以及 32 位宽度的读写访问。

LGT8F88D/168D E2PCTL 控制器结构图



E2PCTL 模拟 E2PROM 功能访问数据 FLASH 空间时，可以支持 8 位、32 位读写宽度。访问程序 FLASH 空间时，支持页擦除和 32 位数据读写。由于 LGT8FX8P 内部 FLASH 的最小存储单元为 32 位，因此建议用 32 位访问方式，特别是对于写操作。32 位访问的读写操作不仅效率高，也有利于保护 FLASH 存储单元的擦写寿命。

LGT8F328P E2PCTL 控制器结构图



LGT8F328P 内部没有多余的数据 FLASH. 因此，LGT8XM 内核与 E2PCTL 共享内部 32K 字节 FLASH 存储空间。用户可以根据需要，将 32K 字节 FLASH 空间划分为程序空间和数据空间。通过配置 E2PCTL 控制器，可以设置模拟 E2PROM 的空间大小。E2PCTL 使用页交换模式实现模拟 E2PROM 逻辑，算法以页(1K 字节)为单位。因此模拟 1K 字节的 E2PROM 空间，需要占用 2K 字节的 FLASH 空间，以此类推，实现 4K 字节的 E2PROM，需要占用 8K 字节的 FLASH 空间。具体实现方式，请参考 E2PCTL 算法实现的描述。

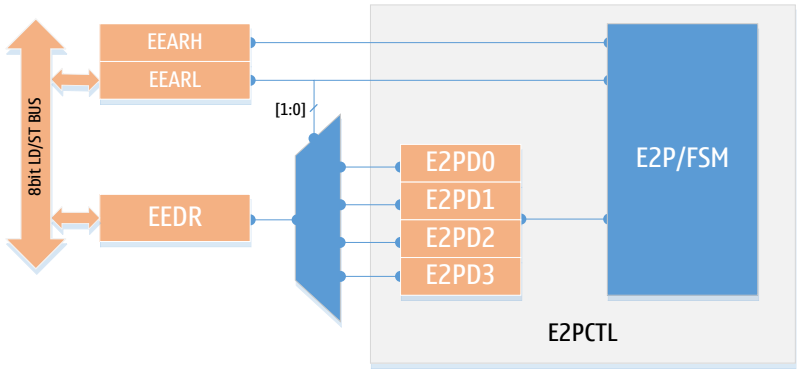
E2PCTL 数据寄存器

E2PCTL 控制器内部有 4 个字节的数据缓存(E2PD0~3)，此 4 字节的缓存组成最终访问 FLASH 空间的 32 位数据接口。

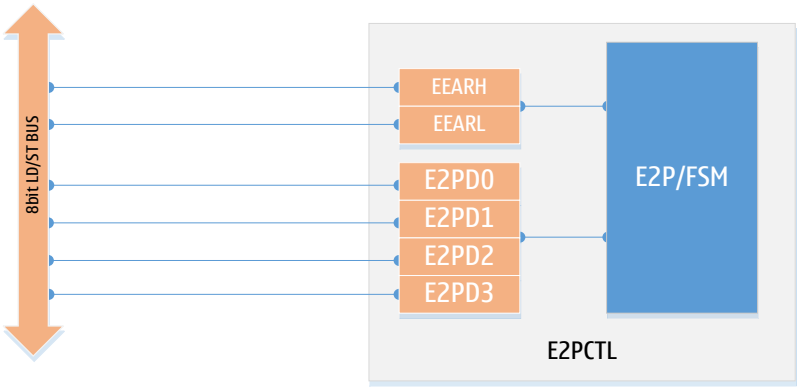
当 E2PCTL 控制器工作在字节读写模式时，EEDR 作为读写字节数据的接口， E2PCTL 更加 EEARL[1:0]的地址信息加载数据到正确的数据缓存中，并根据当前 FLASH 目标地址的数据补齐另外三个字节的数据， 最终将组合的完整 32 位数据更新到 FLASH 中。

当 E2PCTL 工作在 32 位读写模式时， 此时仍然可以使用 EEDR 寄存器作为一个公用的数据接口， 通过 EEARL[1:0]作为地址寻址内部数据缓存，实现读写一个完整的 32 位数据。此外，还可以直接使用数据缓存映射到 IO 空间的寄存器直接访问(E0~3)。

E2PCTL 工作在 8 位字节读写模式时的数据访问示意图：



E2PCTL 工作在 32 位字读写模式时的数据访问示意图：



字节模式用于向下兼容 LGT8FX8D 的字节读写模式。LGT8FX8P 的内置 FLASH 为 32 位接口宽度， 使用 32 位读写模式将给读写效率和 FLASH 的擦写寿命带来极大的好处，因此建议使用 32 位读写模式。

E2PCTL 模拟 E2PROM 接口算法

我们知道，FLASH 存储器在写之前必须先擦除，而擦除操作是以页面为单位的。LGT8FX8P 内置 FLASH 存储器一个页面的大小为 1K 字节。因此为了更新页面中的一个字节数据， 也需要首先擦除掉整个页面的数据，然后更新目标地址数据，并同时恢复页面中其他字节的数据，整个操作不仅仅耗时， 也同时带来因电源意外丢失数据风险。

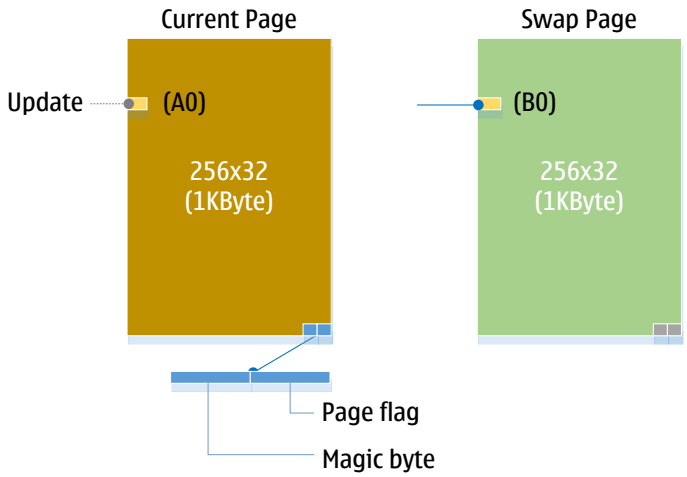
E2PCTL 内部采用页交换算法实现模拟 E2PROM。页交换算法模式可以保证在执行页擦除操作时，不会因为掉电等意外情况导致原有数据的丢失。同时也交换算法使用 2 个页面空间

互为交换的方式交替使用， 也增加了模拟 E2PROM 空间的使用寿命。

在效率方面， E2PCTL 控制器实现了一种连续数据更新模式，减少了因反复更新数据带来的重复擦写过程。

在实现方面， E2PCTL 对每一个页面单独管理，并占用一个页面最后 2 个字节作为页面状态的信息。因此用户在使用大于 1K 的 E2PROM 模拟空间时，需要注意地址跨过 1K 空间的特殊处理。因为每 1K 空间的最后 2 个字节保留给 E2PCTL 使用，用户无法对这 2 个字节的空间进行正常的读写。

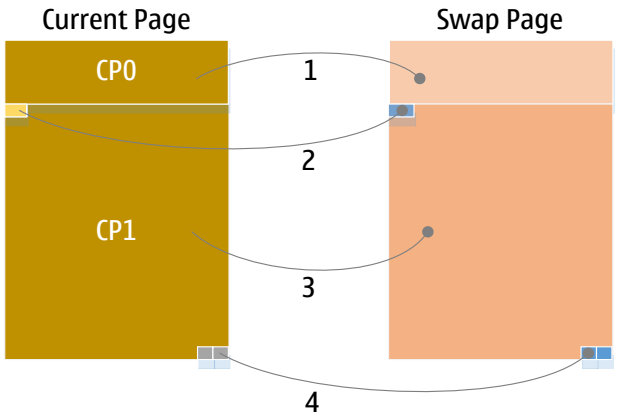
下图为 E2PCTL 基于页交换算法的逻辑示意图：



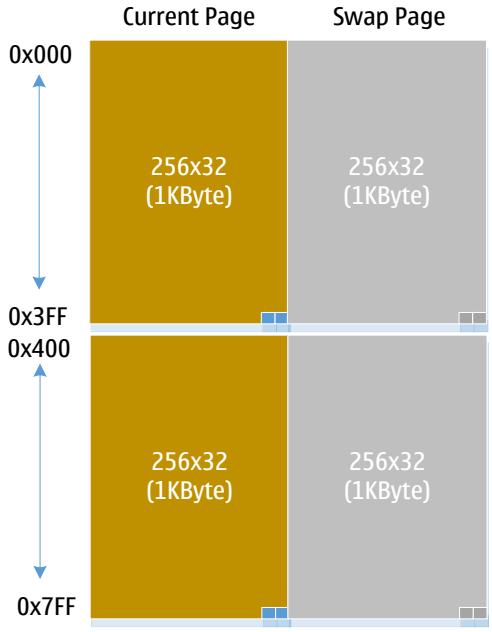
如图所示，E2PCTL 内部使用 2 个页面模拟一个页面大小的 E2PROM 空间。这两个页面一个被标记为当前页面，另外为交换页。E2PCTL 使用页面最后 2 个字节存储页面信息。当我们需要更新页面中的某一个字节时，比如上图中的 A0 字节。首先，我们不会擦除当前页面，而是擦除交换页。然后将当前页面分为 3 个部分操作。首先是在 A0 之前的数据，我们把这部分空间成为 CP0，然后是 A0 之后的数据，这部分空间为 CP1。E2PCTL 会根据用户配置，将 CP0 对应的数据复制到交换页的对应地址，然后将需要更新的数据写到交换页对应的地址上(B0)，最后是复制 CP1 的数据到交换页。

完成上述操作后，数据已经完成交换，但并没有更新页面状态。因此如果在此之前发生掉电或者其他异常，本次更新操作因为并没有完成，之前的数据并不会被破坏，保证了数据的完整性。如果一切顺利，E2PCTL 会在 CP1 交换数据的最后，将更新的页面状态写到之前的交换页面的页面信息中，实现当面页面的更换。此后，交换页面成为当前页。

E2PCTL 页面交换过程如下图所示(1->2->3->4)：



当系统配置的 E2PROM 模拟空间大于 1K 时，E2PCTL 还是以页面为最小单位实现 E2PROM 空间的模拟算法。比如如果用户配置了 2K 的 E2PROM 区域，实际上 E2PCTL 将会占用 4 个页面（4K）的空间。其中 2 个页面为一组，用于实现模拟一个页面大小的 E2PROM 空间。



需要注意的是用户配置的 2K 字节的 E2PROM 空间并不连续，因为每个页面的最后 2 个字节将会被用来保存页面状态信息。

E2PCTL 连续编程模式

由于通过 E2PCTL 更新会导致页面交换，页面交换过程中将会对交换页进行擦除，页擦除不仅耗时，也同时会增加 FLASH 寿命的损耗。因此 E2PCTL 增加了连续写模式。在连续写模式中，用户可以连续更新 E2PROM 区域，只有在连续地址的最后，才会进行页交换操作，对于需要连续更新一整块数据的应用， 连续模式更加有效。

连续编程模式 E2PCTL 控制寄存器 ECCR 的 SWM 位使能。连续模式使能后，后续的写操作将直接将数据写到交换页对应的地址上，在 SWM 模式下， 写操作不会执行 CP0/1 区域数据复制操作。在写最后一个字节前，软件通过 SWM 禁止连续模式，然后执行写， 此后 E2PCTL 将执行完整的 CP0/1 复制操作，并更新页面状态信息。

E2PCTL 读写 FLASH 程序空间

通过 E2PCTL 控制器，可以实现对程序 FLASH 空间的读写访问。与模拟 E2PROM 不同的是，通过 E2PCTL 对程序 FLASH 空间的访问完全需要软件控制。步骤如下：

1. 擦除目标页面，更新数据前需要首先擦除目标页面，页面地址通过 EEAR 寄存器给出。对 FLASH 页面的擦除命令控制，请参考 EECR 寄存器的定义；
2. 写程序 FLASH 空间必须以 32 位为最小单位。通过 E2PD0~3 设置数据；
3. 目标地址由 EEAR 寄存器给出，地址 EEAR[1:0]将会被忽略；

通过 E2PCTL 读写程序 FLASH 空间，可以实现在线程序更新(IAP)功能，在一些需要现场更新应用数据以及需要提供产品自定义更新的应用中，非常有用。

E2PCTL 接口操作流程

E2PCTL 控制器是工作主要通过 4 个寄存器实现，分别为 E2PCTL 控制状态寄存器 EECR、ECCR；数据寄存器 EEDR(E2PD0~E2PD3)以及地址寄存器 EEAR(EEARL/EEARH)。

ECCR 寄存器用于设置 E2PCTL 的工作状态，大部分状态需要在 E2PCTL 工作前设置完成，这个过程一般在系统初始化过程中实现。ECCR 寄存器中的 SWM 位用于使能连续写模式，这个控制位需要在实现连续写操作过程中设置。

EECR 寄存器用于控制选择操作类型，用于选择操作指令，比如设置读、擦除命令。

EEDR 寄存器用于 8 位字节模式接口，E2PD0~3 用于 32 位模式的读写操作；

EEAR 寄存器用于设置读，写的目标地址，也用于设置页擦除操作的页地址。页地址是已页位单位对齐的，一页的大小为 1K 字节，需要注意 EEAR 指定的地址是字节地址。

通过 E2PCTL 接口访问 FLASH 程序空间：

通过 E2PCTL 接口可以实现对 FLASH 程序空间的读写和擦除。对 FLASH 空间的读写仅支持 32 位访问宽度。擦除操作以页位单位，每页的大小的 1K 字节(256x32)。

在写 FLASH 程序空间之前， 首先擦除目标地址所在的页面。E2PCTL 写 FLASH 程序空间不支持连续模式， 用户需要按顺序完成写操作。以下为擦写 FLASH 程序空间的流程：

1. 程序 FLASH 页擦除操作

- 设置 EEAR[14:0]为需要擦除的目标页地址，程序 FLASH 一页大小为 1K 字节，因此 EEAR[14:10]将作为页地址，EEAR[9:0]设置为 0
- 设置 EEPM[3:0] = 1X01，其中 EEPM[2]可设置为 0 或 1
- 设置 EEMPE = 1， 同时 EEPE = 0
- 在四个周期内，设置 EEPE = 1，启动程序 FLASH 擦除流程

2. 程序 FLASH 编程操作

- 写 E2PD0~3，准备 32 位编程数据
- 设置 EEAR 为目标地址，此处地址为 4 字节对齐
- 设置 EEPM[3:0] = 1X10， 其中 EEPM[2]可设置为 0 或 1
- 设置 EEMPE = 1，同时 EEPE = 0
- 在四个周期内，设置 EEPE = 1，启动 FLASH 编程流程

通过 E2PCTL 接口访问 E2PROM 模拟空间：

E2PCTL 控制器通过模拟 E2PROM 接口逻辑访问数据 FLASH 空间。模拟 E2PROM 支持 8 位、16 位以及 32 位数据宽度的读写访问。8 位字节模式对 E2PROM 接口具有更好的兼容性。32 位模式有利于提高存储效率和 FLASH 的使用寿命，因此 32 位读写模式为建议的读写模式。E2PROM 模拟接口支持连续读写模式，在需要一次更新多个连续地址的数据应用中，优势明显，建议采用。

对于 LGT8F88P/168P，数据 FLASH 为独立的存储空间。无需通过 ECCR 寄存器配置和使能 FLASH 数据空间。LGT8F328P 并没有独立的数据 FLASH 空间，数据 FLASH 与程序 FLASH 共享 32K 字节 FLASH 空间。需要通过 ECCR 寄存器使能数据 FLASH 分区功能，并通过 ECCR 寄存器的 ECS[1:0]位配置数据 FLASH 的大小。配置生效后，其他使用方法与 LGT8F88P/168P 相同。

FLASH 控制器在实现 E2PROM 接口时，内部已经实现了在必要时自动擦除数据 FLASH 的逻辑，所以 EPROM 擦除命令是可选的，这个命令只在用户需要单独执行擦除时使用。

EECR 寄存器控制 FLASH 的擦/写时序，包括程序 FLASH 和 E2PROM。具体的操作类型需要通过 EECR 寄存器的 EEPME 和 EEPM[3:0]设定。对 E2PROM 的读操作比较简单，在设置好目标地址和模式后，写 EERE 位即将目标地址对应的 32 位数据读入 FLASH 控制器内部，用户可以通过 EEDR 寄存器读取感兴趣的字节。FLASH 控制器并没有实现对程序 FLASH 空间的读操作，用户可以方便的使用 LPM 或者通过程序 FLASH 在数据统一映射空间的地址处使用 LD/LDD/LDS 指令读取。

1. 8 位模式，编程 E2PROM

- 设置目标地址到 EEARH/L 寄存器
- 设置新的数据到 EEDR 寄存器
- 设置 EEPM[3:1] = 000，EEPM[0]可设置为 0 或 1
- 设置 EEMPE = 1， 同时 EEPE = 0
- 在四个周期内，设置 EEPE = 1

当设置完成后，FLASH 控制器将启动编程操作，编程期间 CPU 将保持在当前的指令地址上，直到操作完成后才会继续运行。在编程过程中，如果需要擦除数据 FLASH，FLASH 控制器将会自动启动擦除流程。

2. 32 位模式，编程 E2PROM

- 通过 E2PD0~3，准备 32 位数据
- 设置目标地址到 EEARH/L 寄存器。注意这里是字节对齐的地址，FLASH 控制器用 EEAR[15:2]作为访问 FLASH 的地址。
- 设置 EEPM[3:1] = 010，EEPM[0]可设置为 0 或 1
- 设置 EEMPE = 1， 同时 EEPE = 0
- 在四个周期内，设置 EEPE = 1

3. 8 位模式，读 E2PROM

- 设置目标地址到 EEARH/L 寄存器
- 设置 EEPM[3:1] = 000
- 设置 EERE = 1 启动 E2PROM 读操作
- 等待 2 个周期 (执行两个 NOP 操作)
- 目标地址对应的数据被更新到 EEDR 寄存器

4. 32 位模式，读 E2PROM

- 设置 EEARH/L 为目标地址，地址为 4 字节对齐
- 设置 EEPM[3:1] = 010，开启 32 位接口模式
- 设置 EERE = 1，启动 E2PROM 读操作
- 等待 2 个系统时钟周期 (执行两个 NOP 指令)

E2PCTL 访问模拟 E2PROM 空间，支持连续编程模式，连续访问模式对于需要一次更新一个数据块的应用非常高效，也有利于提高 FLASH 的使用寿命。连续编程模式仅支持 32 位宽度的数据编程操作。

连续访问模式通过 ECCR 寄存器的 SWM 位使能。SWM 使能后，接下来通过 E2PCTL 写模拟 E2PROM 空间的操作都在连续编程模式。在连续编程模式下，E2PCTL 控制器会根据目标地址内的数据情况自动处理换页。但在连续编程模式过程中如果发生换页，控制器在连续编程过程中，不会自动将 CP0/1 区域的数据交换，也不会更新页面信息。

当连续编程到最后一次操作前，通过清零 SWM 位关闭连续编程模式，然后在非 SWM 模式下启动最后一次编程操作，编程结束后，E2PCTL 会自动将 CP0/1 区域的数据复制到交换页，并更新交换页的信息，使之成为当前有效页，从而完成整个连续编程操作。

5. 连续编程模式操作流程：

1. 通过 ECCR 配置数据 FLASH 的大小，并使能 SWM 位
2. 使用 32 位模式编程模拟 E2PROM 区域
3. 如果不是最后一次操作，回到步骤 2 继续编程下一个数据
4. 如果达到最后一次编程，首先通过 SWM 禁止连续编程模式，然后使用步骤 2 的操作流程完成最后一次编程

E2PCTL 高效 FLASH 数据管理

E2PCTL 控制器除了实现连续编程模式，也可以通过 ECCR 寄存器的 CP0/1 位对页交换过程数据交换复制进行独立控制。ECCR 寄存器的 CP0/1 分别用于控制页交换过程中对于当前页面中 CP0/1 区域数据的交换操作。清零 CP0/1 位，在页交换过程中将不会交换当前页中对应区域的数据。本节提供的一种高效管理方法，将会利用这一特性。

在 FLASH 数据更新过程中，最为耗时的操作发生在交换页擦除过程。因此我们可以寻址一种最大限度减小页擦除次数的数据管理方法，既能提高编程效率，也能减少寿命损耗。

这里我们提供一种参考算法，适用于基于数据块数据管理应用：

1. 假定用户数据只是一个完整的数据块，数据块大小 4 字节的整数倍；
2. 每次数据更新将会更新一个完整的数据块
3. 数据块信息除了存放用户数据， 还需要存放一个块管理信息

以上三个条件下，我们可以充分利用 E2PCTL 的连续编程模式和自动页交换机制，实现一个高效率的 FLASH 数据管理方法。

由于是每次更新的数据为一个相同大小的数据块，并且每块数据结构中保存有指向下一块数据的地址信息，因此我们可以每次更新数据时按地址顺序编程 FLASH，无需做 CP0/1 的数据复制。同时由于每次都是更新数据到一个已擦除的区域， 也不会发生页擦除。

当数据写完最后一块，其结构信息指向的下一块数据区回到页的起始地址。此后再发生数据写操作，E2PCTL 将会启动一次页擦除过程，并更新当前活动页面。

FLASH 操作的保护措施

如果 VCC 电压偏低，FLASH 的擦写操作可能会因为电压太低而发生错误。

FLASH/数据在低压下的擦写操作错误可能由两种原因。首先，正常的 FLASH 擦写操作需要一个最小工作电压，低于这个电压，操作将会失败而导致数据发生错误。第二个原因，是内核运行在某一频率下，也同样需要一个最小电压要求，当低于这个电压，将会导致指令执行出错，从而使得 FLASH 的操作发生错误。

可以通过下面简单的方法避免类似问题：

在供电电压较低时，让系统进入复位状态。这可以通过配置内部的低压检测电路(VDT)实现。如果 VDT 检测到当前的工作电压低于设置的阀值，VDT 将会输出一个复位信号。如果 VDT 的阀值不能满足应用的需要，可以考虑在外部增加一个复位电路。

寄存器描述

FLASH 地址寄存器- EEARH/EEARL

EEARH/EEARL		
EEARH: 0x22 (0x42) EEARL: 0x21 (0x41)		默认值: 0x0000
bits	EEAR[15:0]	
R/W	R/W	
位定义		
[7:0]	EEARL	EFLASH/E2PROM 访问地址低 8 位。
[14:8]	EEARH	EFLASH/E2PROM 访问地址高 7 位
[15]	-	保留不用

当使用 E2PCTL 控制器访问程序 FLASH 区域时，EEAR[14:2]用作访问以 4 字节对齐的整个程序空间。EEAR[1:0]只在访问数据寄存器 EEDR 时使用。具体请参考下面关于 EEDR 数据寄存器的描述。E2PCTL 控制器支持 8/16/32 位模式，无论是哪一种模式，此处的 EEAR 都是以字节对齐寻址。

FLASH 数据寄存器- EEDR/E2PD0

EEDR/E2PD0 – FLASH/E2PROM 数据寄存器 0		
EEDR/E2PD0: 0x20 (0x40)		默认值: 0x00
bits	EEDR[7:0]	
R/W	R/W	
位定义		
[7:0]	EEDR E2PD0	E2PCTL 数据寄存器 16/32 位模式时，用于存取最低字节

FLASH 数据寄存器- E2PD1

E2PD1 – E2PCTL 数据寄存器 1	
E2PD1: 0x5A	默认值: 0x00
bits	E2PD1[7:0]

R/W	R/W	
位定义		
[7:0]	E2PD1	16 位模式时用于存储 16 位数据的高 8 位 32 位模式时用于存储低 16 位数据的高 8 位

FLASH 数据寄存器- E2PD2

E2PD2 – FLASH 数据寄存器 2		
E2PD2: 0x57		默认值: 0x00
Bits	E2PD2[7:0]	
R/W	R/W	
位定义		
[7:0]	E2PD2	32 位模式时用于存储高 16 位数据的低 8 位

FLASH 数据寄存器- E2PD3

E2PD3 – FLASH 数据寄存器 3		
E2PD3: 0x5C		默认值: 0x00
Bits	E2PD3[7:0]	
R/W	R/W	
位定义		
[7:0]	E2PD3	32 位模式时用于存储高 16 位数据的高 8 位

FLASH 模式控制寄存器- ECCR

ECCR – FLASH/E2PROM 配置寄存器								
ECCR: 0x36 (0x56)					默认值: 0x0C			
bits	WEN	EEN	ERN	SWM	CP1	CP0	ECS1	ECS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	1	1	0	0
位定义								

[7]	WEN	ECCR 写使能控制 在修改 ECCR 前，必须先将 WEN 写 1，然后在 6 个系统周期内，更新 ECCR 寄存器的内容
[6]	EEN	E2PROM 使能，仅对 LGT8F328P 有效 1：使能 E2PROM 模拟，将会从 32KFLASH 中保留部分空间 0：禁用 E2PROM 模拟，32KFLASH 全部用于程序空间
[5]	ERN	写 1 将复位 E2PCTL 控制器
[4]	SWM	连续写模式，适用于模拟 E2PROM 控制器操作
[3]	CP1	页交换 CP1 区域使能控制
[2]	CP0	页交换 CP0 区域使能控制
[1:0]	ECS[1:0]	E2PROM 空间配置 00：1KB E2PROM, 30KB 程序 FLASH 01：2KB E2PROM, 28KB 程序 FLASH

		10：4KB E2PROM, 24KB 程序 FLASH 11：8KB E2PROM, 16KB 程序 FLASH

FLASH 访问控制寄存器- EECR

EECR – FLASH/E2PROM 控制寄存器								
EECR: 0x1F (0x3F)				默认值: 0x00				
bits	EEPM3	EEPM2	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0
位定义								

[7:4]	EEPM[3:0]	EFLASH/EPROM 访问模式控制位				
		[3]	[2]	[1]	[0]	模式说明
		0	0	0	x	8 位模式读/写 E2PROM（默认）
		0	0	1	x	16 位模式读/写 E2PROM
		0	1	0	x	32 位模式读/写 E2PROM
		1	x	0	0	E2PROM 擦除(可选操作)
		1	x	0	1	程序 FLASH 擦除(页擦除)
		1	x	1	0	程序 FLASH 编程
		1	x	1	1	复位 FLASH/E2PROM 控制器
[3]	EERIE	FLASH/E2PROM 就绪中断使能控制。写 1 使能，写 0 禁止。当 EEPE 被硬件自动清零后，E2PROM 就绪中断有效。在 EPROM 操作过程中，将不会产生这个中断				
[2]	EEMPE	FLASH/E2PROM 编程操作使能控制位 EEMPE 用于控制 EEPE 是否有效，当同时设置 EEMPE 为 1，EEPE 为 0 后，在之后的四个周期内，设置 EEPE 为 1 将启动编程操作。否则编程操作无效。四个周期后，EEMPE 被自动清零				
[1]	EEPE	FLASH/E2PROM 编程操作使能位				
[0]	EERE	E2PROM 读使能位，数据将在两个系统周期以后有效				

通用 I/O 寄存器- GPIOR2

GPIOR2 – 通用 I/O 寄存器 2		
GPIOR2: 0x2B (0x4B)		默认值: 0x00
Bits	GPIOR2[7:0]	
R/W	R/W	
初始值	0x00	
位定义		
[7:0]	GPIOR2	通用 I/O 寄存器 2，用于存储用户自定义数据

通用 I/O 寄存器- GPIOR1

GPIOR1 – 通用 I/O 寄存器 1		
GPIOR1: 0x2A (0x4A)		默认值: 0x00
Bits	GPIOR1[7:0]	
R/W	R/W	
初始值	0x00	
位定义		
[7:0]	GPIOR1	通用 I/O 寄存器 1，用于存储用户自定义数据

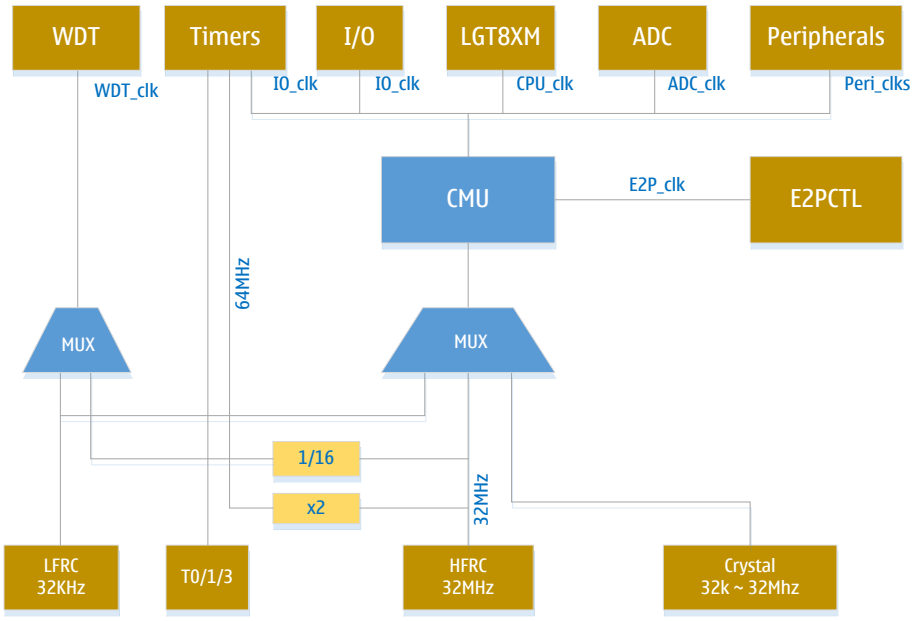
通用 I/O 寄存器- GPIOR0

GPIOR0 – 通用 I/O 寄存器 0		
GPIOR0: 0x1E (0x3E)		默认值: 0x00
Bits	GPIOR0[7:0]	
R/W	R/W	
初始值	0x00	
位定义		
[7:0]	GPIOR0	通用 I/O 寄存器 0，用于存储用户自定义数据

系统时钟与配置

系统时钟分布

LGT8FX8P 支持多种时钟输入。系统可以工作在三种主要的时钟源，分别是内部 32KHz 可校准 RC 振荡器，内部 32MHz 可校准 RC 振荡器以及外部 400KHz ~ 32MHz 晶振输入。下图为 LGT8FX8P 时钟系统分布，CMU 是整个时钟管理的中心，负责系统时钟的分频，为不同的模块产生独立的时钟以及对时钟进行控制等等。一般的应用中，并不不要全部的时钟同时工作，为了减小系统功耗，系统功耗管理根据不同的休眠模式，关闭没有使用的模块时钟。具体操作细节，请参考功耗管理相关章节。



CPU_clk

用于驱动 LGT8XM 内核以及 SRAM 的运行。比如驱动通用工作寄存器，状态寄存器等。CPU 时钟停止后，内核将不会继续执行指令和进行计算。系统执行 SLEEP 指令进去休眠模式后，内核时钟将会被关闭。

Peri_clk

用于驱动大部分外设模块，比如定时/计数器，SPI，USART 等。IO 时钟也用于驱动外部中断模块。当外设时钟因休眠而停止后，某些可以用了唤醒系统的外设部分工作在独立的时钟或异步模式。比如 TWI 的地址识别功能可以唤醒大部分休眠模式，此时的地址识别部分工作在异步模式。

E2P_clk

E2P_clk 时钟用于产生 FLASH 接口访问时序。E2P_clk 产生访问 E2PCTL 访问 FLASH 接口的时序。E2P_clk 固定来自内部 32MHz HFRC 振荡器的 32 分频(1MHz)。如果用户需要使用 E2PCTL 模块读写内部程序 FLASH 或者数据 FLASH 空间，需要提前使能内部 32MHz 振荡器。

Asy_clk

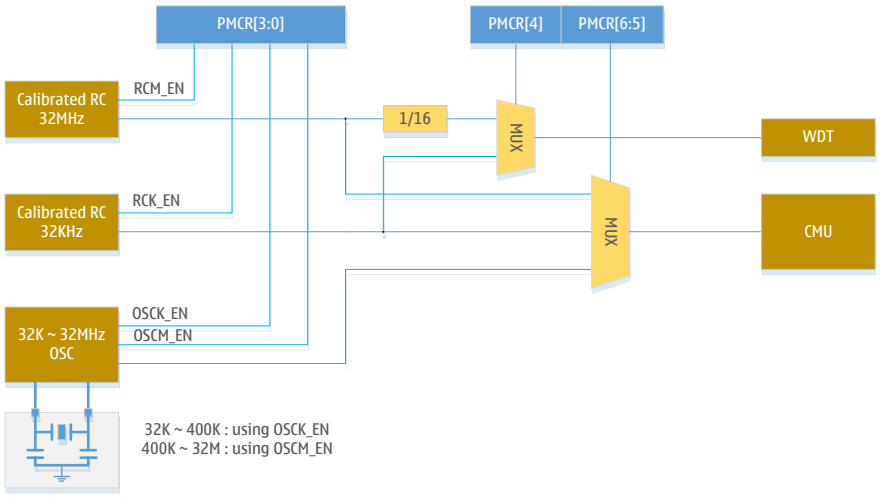
异步定时器时钟。定时/计数器可以直接使用外部时钟或晶振(32.768K)驱动。这种独立的时钟模式，可以在系统处理休眠模式时，定时器仍然保持运行。

WDT_clk

内部看门狗定时器时钟源，可以配置选择内部 32KHz LFRC 振荡器，或者来自内部 32MHz HFRC 的 16 分频（2MHz）。系统上电后，看门狗默认时钟源为 32KHz LFRC 振荡器。

时钟源选择

LGT8FX8P 支持 4 种时钟源输入，用户可以通过 PMCR 寄存器实现对时钟源的使能控制以及完成主时钟的切换。下面是 PMCR 的控制结构图：



LGT8FX8P 内部 OSC 振荡器可以工作在高频和低频两种模式下，用户需要根据外接晶振的实际大小控制内部 OSC 振荡器工作在正确的模式下。同样内部的 RC 振荡器也分为高频和低频两种。PMCR 寄存器的最低 4 位用于控制这四种时钟源。控制关系如下：

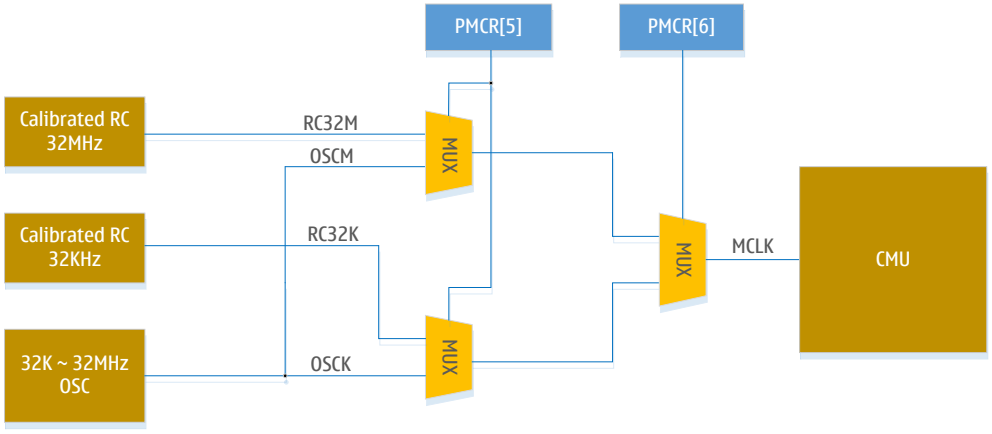
PMCR	对应时钟源
PMCR[0]	32MHz RC 使能控制，1 使能，0 关闭
PMCR[1]	32KHz RC 使能控制，1 使能，0 关闭
PMCR[2]	400K ~ 32MHz OSC 模式使能，1 使能，0 关闭
PMCR[3]	32K ~ 400K OSC 模式使能， 1 使能，0 关闭

LGT8FX8P 系统上电后，默认使用 32MHz RC 作为系统时钟源，内核工作在时钟源的 8 分频(4MHz)。用户可以通过设置 PMCR 寄存器以及系统预分频寄存器(CLKPR)改变默认配置。

如果用户需要更改主时钟源配置，需要在切换时钟前保证切换后的时钟源处于稳定的工作状态。因此需要在切换主时钟源之前，通过 PMCR[3:0]使能所需时钟源，并等待到时钟稳定后才能进行切换。

当用户切换主时钟到外部晶振时，虽然用户使能了外部晶振，但也不排除因配置错误或晶振失效导致晶振无法起振。如果在此时切换到外部晶振，切换后系统将停止工作。因此，从系统可靠性考虑，建议打开看门狗定时器，从软件设计的角度避免此类问题。

时钟源使能并等待稳定后，可以通过 PMCR[6:5]切换主时钟。其中 PMCR[5]用于选择是内部 RC 振荡器和外部晶振，PMCR[6]用于选择高速时钟源和低速时钟源。



主时钟源选择：

PMCR[6]	PMCR[5]	主时钟源
0	0	内部 32MHz RC 振荡器(系统默认)
0	1	外部 400K ~ 32MHz 高速晶振
1	0	内部 32KHz RC 振荡器
1	1	外部 32K ~ 400KHz 低速晶振

时钟源控制时序

为保护 PMCR 寄存器被意外修改，对 PMCR 寄存器的修改需要严格安装指定的时序进行。PMCR 寄存器的最高位(PMCR[7])用于实现时序控制。用户在修改 PMCR 其他位之前，必须首先要将 PMCR[7]置 1，在置 1 操作后的 6 个周期内，更改 PMCR 其他寄存器的值。6 个周期之后，对 PMCR 的直接修改将失效。

下面以切换到外部高速晶振为例，列出建议的操作步骤：

(1) 使能时钟源

- 设置 PMCR[7] = 1
- 在六个周期内，设置 PMCR[2] = 1，使能外部高速模式外部晶振
- 等待外部晶振稳定(等待时间因晶振不同而不同，一般 us 级等待即可)

(2) 切换主时钟源

- 设置 PMCR[7] = 1
- 在六个周期内，设置 PMCR[6:5] = 01，系统将工作时钟自动切换至外部晶振
- 执行几个 NOP 操作，提高稳定性(可选操作)

[注意]：在以上切换主时钟的操作中，要保证当前系统时钟正常工作，在切换到外部晶振以后，才可以关闭之前的内部 RC 振荡器。

系统时钟预分频控制

LGT8FX8P 内部有一个系统时钟预分频器，可以通过时钟预分频寄存器(CLKPR)进行控制。这种功能可以用于当系统不需要非常高的处理能力时，减小系统功耗。预分频设置对系统支持的时钟源都有效。时钟预分频能够影响到内核执行时钟以及所以同步外设。

当在不同的时钟预分频设置之间切换时，系统时钟预分频确保在切换过程中不会产生毛刺，而已会保证不会有过高频的中间状态。分频切换是立即执行的，当寄存器改变生效后，最多在 2~3 个当前系统时钟周期后，系统时钟就切换到了新的分频时钟。

为了避免对时钟分频寄存器的误操作，对 CLKPR 的修改也必须遵循一个特殊的时序流程：

- 设置时钟预分频更改使能位(CLKPCE)为 1，CLKPR 其他所以位为 0
- 在四个周期内，把需要的值写入 CLKPS，同时 CLKPCE 写 0

在更改时钟预分频寄存器前， 需要禁止中断功能，以保证写时序能够完整的进行。
关于主时钟预分频寄存器 CLKPR 的具体定义，请参考本章节寄存器描述部分。

内部 RC 振荡器校准

LGT8FX8P 内部包含两个可校准 RC 振荡器，经过校准后，均可达到±1%以内的精度。其中 32MHz RC 默认用于系统工作时钟。

LGT8FX8P 出产前，内部 32MHz HFRC 和 32KHz LFRC 都进行了校准，并把校准值写入系统配置信息区域。系统省电过程中，这些校准值将会被读入到内部寄存器中，通过寄存器实现对 RC 频率的重新校准。

校准寄存器位于 IO 地址空间，用户程序可以读写。对于频率有特殊需求的应用，可以通过修改校准寄存器方式调整内部振荡器的频率输出。修改校准寄存器不会改变出厂配置信息，系统重新上电或者用户启动的配置位重新加载操作，校准寄存器将会恢复到出厂设置。

寄存器定义

32MHz HFRC 振荡器校准寄存器- RCMCAL

RCMCAL – 32MHz HFRC 校准寄存器		
RCMCAL: 0x66		默认值：出厂配置
Bits	RCCAL[7:0]	
R/W	R/W	
位定义		
[7:0]	RCCAL	系统上电后，寄存器的值将被系统配置信息中的 RC 校准值替换。

32KHz RC 振荡器校准寄存器- RCKCAL

RCKCAL – 32MHz RC 校准寄存器		
RCKCAL: 0x67		默认值: 出厂设置
Bits	RCKCAL[7:0]	
R/W	R/W	
位定义		
[7:0]	RCKCAL	将校准值写入 RCKCAL 寄存器完成对 32KHz RC 振荡器的校准

时钟源管理寄存器- PMCR

PMCR – 时钟源管理寄存器							
PMCR: 0xF2				默认值: 0x03			
Bits	PMCE	CLKFS/CLKSS	WCLKS	OSCKEN	OSCMEN	RCKEN	RCMEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义							

[0]	RCMEN	内部 32MHz RC 振荡器使能控制，1 使能，0 禁止
[1]	RCKEN	内部 32KHz RC 振荡器使能控制，1 使能，0 禁止
[2]	OSCMEN	外部高频晶振使能控制，1 使能，0 禁止
[3]	OSCKEN	外部低频晶振使能控制，1 使能，0 禁止
[4]	WCLKS	WDT 时钟源选择， 0 – 选择内部 32MHz HFRC 振荡器的 16 分频 1 – 内部 32KHz LFRC 振荡器
[5]	CLKSS	主时钟源选择控制，选择时钟源类型，请参考时钟源选择部分
[6]	CLKFS	主时钟源频率控制，选择时钟频率类型，请参考时钟源选择部分
[7]	PMCE	PMCR 寄存器更改使能控制位。 在更改 PMCR 其他位置之前，必须首先设置此位，然后在四个周期内设置其他位的值。

主时钟预分频寄存器- CLKPR

CLKPR – 主时钟预分频寄存器								
CLKPR: 0x61					默认值: 0x03			
Bits	WCE	CKOEN1	CKOEN0	-	PS3	PS2	PS1	PS0
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
位定义								

[3:0]	CLKPS	时钟预分频选择位				
		PS3	PS2	PS1	PS0	分频参数
		0	0	0	0	1
		0	0	0	1	2
		0	0	1	0	4
		0	*0*	*1*	*1*	*8(默认配置)*

		0	1	0	0	16
		0	1	0	1	32
		0	1	1	0	64
		0	1	1	1	128
		1	0	0	0	256
		其他值				保留
[4]	-	保留不用				
[5]	CKOEN0	设置系统时钟是否在 PB0 引脚上输出				
[6]	CKOEN1	设置系统时钟是否在 PE5 引脚上输出				
[7]	WCE	时钟预分频更改时钟控制 在改变 CLKPR 寄存器的其他位之前，必须首先单独设置 CKWEN 为 1，然后在之后的四个系统周期内，对其他位进行设置。四个周期结束后，CKWEN 自动清零。				

功耗管理

概述

休眠模式通过关闭系统时钟以及时钟模块，从而减小系统功耗。LGT8FX8P 提供了非常灵活多样的休眠模式和模块控制器，用户可以根据应用，实现最理想的低功耗配置。

LGT8FX8P 在进入休眠模式时，并不会自动关闭模拟功能模块，比如 ADC，DAC，比较器(AC)，低电压复位模块(LVD)等等，软件需根据应用要求，在进入休眠前关闭不需要的模拟功能，并在系统唤醒后恢复正确的状态。

LGT8FX8P 支持多种休眠模式，其中包括 ADC 专用的噪声消除模式，用于消除 ADC 转换过程中数字部分对 ADC 电源的干扰。除此之外，其他均为功耗控制模式，共分为五种：

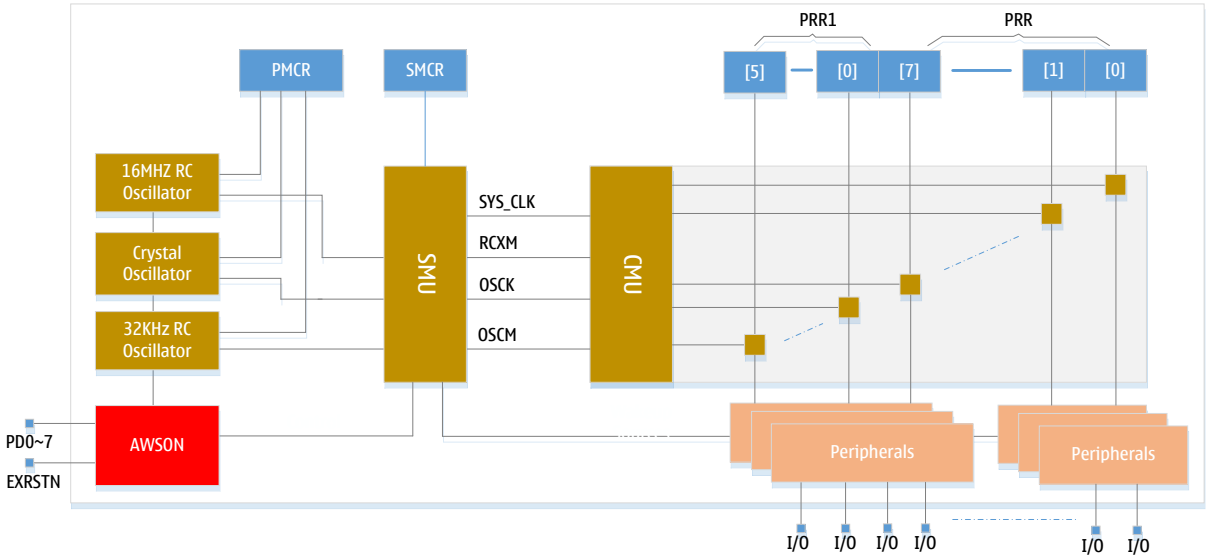
休眠模式	功能说明
空闲模式(IDLE)	仅仅关闭内核时钟，其他外设模块正常工作，所有有效中断源均可以将内核唤醒
省电模式(Save)	与 DPS0 模式相同，Save 模式为与 LGT8FX8D 保持兼容
掉电模式(DPS0)	与 Save 模式相同，支持唤醒源包括： ● 所有引脚电平变化 ● 看门狗定时唤醒 ● 异步模式的 TMR2 唤醒
掉电模式(DPS1)	关闭所有内外部振荡器，支持唤醒源包括： ● 所有引脚外部电平变化 ● 外部中断 0/1 ● 工作于 32K LFRC 的看门狗定时器
掉电模式(DPS2)	关闭内核电源，最低功耗模式，支持的唤醒源包括： ● 外部复位 ● PORTD 引脚电平变化 ● LPRC 定时唤醒(128ms/256ms/512ms/1s) 需要注意，从 DPS2 唤醒的过程与上电复位相同

LGT8FX8P 支持深度休眠 DPS2，在该模式下，系统内部 LDO 处于掉电状态，内核寄存器，所有外设控制器以及 SRAM 等均处于掉电状态， 其中的数据将不会保持。FLASH 存储单元也将处于掉电状态，因此 DPS2 模式能够达到系统最小功耗。掉电模式可以通过端口 D(PORTD)引脚电平变化唤醒，也可以选择 5 级定时唤醒。用于唤醒的 DPS2 的定时器由于不支持校准，精度在 15%左右，只适合用于精度较低的定时唤醒应用。

系统从 DPS2 模式唤醒， 会首先开启 LDO，这个过程和上电过程相同。芯片将执行完整的上电复位启动过程，加载配置信息，然后从复位向量指向的地址运行程序。

除 DPS2 以外的其他模式，不会关闭内部电源，在休眠过程中，所有寄存器信息以及 RAM 数据均不会丢失。唤醒后，内核从休眠前的最后一条指令继续执行。

系统功耗管理示意图：



如上图所示，LGT8FX8P 主要通过休眠模式控制器(SMU)以及时钟管理单元(CMU)控制整个系统的功耗。从节省功耗的级别上，我们可以把功耗分为 4 个等级:

第一级是通过 PRR 寄存器控制模块工作时钟，通过关闭没有使用模块的时钟，节省系统运行的动态功耗。一般情况下，这种级别能够节省的功耗并不明显。

第二级是通过切换主时钟源到低频时钟上，并关闭没有使用的时钟源模块以及其他模拟模块，这种模式基本上可以得到非常可观的系统运行功耗和休眠功耗。

第三级别是通过让系统进入到掉电模式(DPS1)，DPS1 模式下 LGT8FX8P 可以获得极地的待机功耗，从断电模式唤醒后，软件可以通过 MCUSR 寄存器读取复位前的状态。

第四级别是掉电模式(DPS2)，这个模式将关闭内核电源，可达到最低的系统功耗。因为关闭了内核电源，这种模式下所有数据信息将会丢失。唤醒后立刻执行一个上电复位流程，系统重新开始从复位向量处运行。

AWSON 电源管理

与 LGT8FX8D 相比，掉电模式 DPS2 为一个全新的功耗模式。DPS2 模式用于对休眠功耗有更高要求的应用。进入 DPS2 模式后，系统仅维持一个静态的模块(AWSON)处于工作状态，其他电路均处于完全掉电状态。

AWSON 模块专用于负责 DPS2 模式的休眠和唤醒控制，AWSON 模块主要由 IO 唤醒控制逻辑以及一个低功耗的 LPRC 组成。软件可以通过 IOCWK 寄存器以及 DPS2R 寄存器实现对 AWSON 的控制。

IOCWK 寄存器用于控制 PD0~7 电平变化的唤醒功能。DPS2R 寄存器用于控制 DPS2 模式以及 LPRC 的功能模式。具体信息请参考本节末寄存器定义部分。

使用 DPS2 模式前，软件设置 IOCWK 使能所需唤醒 IO， 或者通过 DPS2R 寄存器使能 LPRC 并配置定时唤醒周期， 然后通过 DPS2R 寄存器的 DPS2EN 位使能 DPS2 模式。设置完成后，软件需要通过 SMCR 寄存器设置 DPS2 休眠模式， 然后执行 SLEEP 指令进入休眠。

休眠模式与唤醒源

LGT8FX8P 支持 5 种休眠模式，用户可以根据应用需求选择合适的休眠模式。SMCR 寄存器包含了休眠模式的控制设置，执行 SLEEP 指令后，内核进入休眠模式。为获得更加理想的休眠功耗，建议在内核进入休眠模式前，关闭所有没有使用的时钟以及模拟模块。但需要注意的是，某些唤醒源的产生需要工作时钟，如果需要使用这类唤醒源，请保持相关时钟源的工作状态。

休眠模式与唤醒方式：

休眠模式	有效时钟				唤醒源							
	内核时钟	外设时钟	ADC 时钟	异步时钟	引脚电平变化	外部中断 0/1	TWI 地址匹配	TMR2 中断	ADC 转换结束	看门狗溢出	外设中断	PD 电平变化
空闲模式(IDLE)		X	X	X	X	X	X	X	X	X	X	X
ADC 噪声抑制			X	X	X	X	X	X	X	X		X
省电模式(SAVE)				X	X	X	X	X		X		X
掉电模式(DPS0) (With RC32K)				X	X	X		X		X		X
掉电模式(DPS1) (Without RC32K)				X	X	X		X				X
掉电模式(DPS2) (Without LDO)												X

如果需要进入以上 5 种休眠模式，SMCR 中的 SE 位必须置 1，使能休眠模式控制。然后执行一条 SLEEP 指令即可。SMCR 中的 SM0/1/2 用于选择不同的休眠模式。具体的信息请参考下面的描述。

在 MCU 处于休眠模式下，如果唤醒源有效，MCU 将会在 4 个周期后被唤醒，继续执行指令。如果中断保持有效，中断也将立即响应，进入中断服务子程序。如果在 SLEEP 模式下发生了系统复位，MCU 也将会被唤醒，并从复位向量开始执行。

当 MCU 处于 Power/Off 模式下，系统可以通过外部中断 INT0/1 唤醒，唤醒后 MCU 将从 sleep 前的位置继续执行。

空闲模式(IDLE)

当 SM2...0 设置为 000，执行 SLEEP 指令后，MCU 进入到 IDLE 模式，IDLE 模式将会关闭掉内核工作时钟，除此之外的其他外设都能正常工作。

IDLE 模式可以通过外部中断以及内部中断等唤醒。如果不需要使用比较器以及 ADC 作为唤醒源，建议将其关闭。

IDLE 模式因为仅仅关闭了内核运行的时钟，所以并不能得到明显的功耗降低。IDLE 模式下，内核也将停止执行和取指令，因此可以降低内部程序 FLASH 的运行功耗。

但 IDLE 模式拥有比较灵活的唤醒方式，用户可以通过降低系统主时钟以及关闭不需要的模块获取更加理想的运行功耗。

ADC 噪声抑制模式

当 SM2...0 设置为 001，执行 SLEEP 指令后，MCU 进入 ADC 噪声抑制模式。此模式下，内核以及大部分外设都将停止工作，ADC，外部中断，TWI 地址匹配，WDT 以及工作在异步时钟模式下的定时/计数器 2 都可以正常工作。

ADC 噪声一直模式主要用于为 ADC 转化提供一个良好的工作环境。降低数字模块对模拟转换的高频干扰。进入这个模式后，ADC 将自动启动采样转换，转换的数据保存到 ADC 数据寄存器后，ADC 转换结束中断将 MCU 从 ADC 噪声模式下唤醒。

省电模式(Save)

当 SM2...0 设置为 010，执行 SLEEP 指令后，MCU 进入到 Save 模式。这种模式下，系统将关闭掉所有模块的工作时钟。此模式因为关闭了所有模块的工作时钟，因此只能通过异步模式唤醒，外部中断，TWI 地址匹配以及工作在独立时钟源模式下的 WDT 都可以产生此模式下的唤醒信号。

此种模式可以关闭除主时钟源以为的所有模块。为实现更加理想的运行功耗，建议在进入此中模式前，将系统主时钟切换到内部 32K RC 或者外部 32KHz 低频晶振，然后关闭掉所以没有被使用的时钟源以及模拟模块。

掉电模式 DPS0

当 SM[2:0]设置为 110，执行 SLEEP 指令后，MCU 将进入到 DPS0 模式。进入 DPS0 后，除内部 32KHz RC 外，其他时钟源均被关闭。此种模式可以通过外部中断 INT0/1 唤醒；如果使能了 WDT 的中断功能，也可以通过 WDT 实现定时唤醒。

掉电模式 DPS1

当 SM[2:0]设置为 011，执行 SLEEP 指令后，MCU 将进入到 DPS1 模式。进入 DPS1 后，系统所有时钟源均被关闭。此种模式可以使用 IO 的电平变化，看门狗唤醒。

掉电模式 DPS2

设置 SM[2:0]为 111， 并通过 DPSR2 寄存器的 DPS2EN 使能 AWSON 模块，执行 SLEEP 指令后将进入 DPS2 模式。进入 DPS2 模式后，系统关闭内核电源。所以寄存器以及 RAM 数据将会丢失。从 DSP2 唤醒过程与上电复位过程相同。

DPS2 模式下， 由于关闭了内核电压，寄存器信息丢失，因此端口的控制状态也将全部恢复到输入状态，所有 IO 的输出驱动以及上拉控制也将关闭。

FLASH 电源控制以及快速唤醒

当系统处于 SLEEP 模式后，内核将不会继续执行指令，此时可以选择关闭 FLASH 的电源，以获得更低的待机功耗。这个功能可以通过 MCUCR 寄存器的 FPDEN 位控制实现；

在掉电模式下，系统可以使用外部中断或者 WDT 唤醒，为了滤除外部信号可能的干扰，内部唤醒电路包含了一个可配置的滤波电路，用户可以根据需要选择合适的滤波宽度。滤波电路的配置可以通过 MCUCR 寄存器的 FWKPEN 实现。

MCUCR[FWKPEN]滤波宽度控制：

FWKPEN	滤波宽度
0	260us (默认)
1	32us

寄存器描述

休眠模式控制寄存器- SMCR

SMCR – 休眠模式控制寄存器						
SMCR: 0x33(0x53)			默认值: 0x00			
Bits			SM2	SM1	SM0	SE
R/W	-		R/W	R/W	R/W	R/W
位定义						
[0]	SE	休眠模式使能控制位，设置为 1 后，执行 SLEEP 指令，内核将进入休眠模式。SE 位可以保护系统意外进入休眠模式。唤醒后，建议立刻清除 SE 位。				
[3:1]	SM	休眠模式选择				
		SM2	SM1	SM0	模式说明	
		0	0	0	IDLE 模式	
		0	0	1	ADC 噪声抑制模式	
		0	1	0	Save 模式	
		0	1	1	DPS1 模式	
		1	1	0	DPS0 模式	
		1	1	1	DPS2 模式	
		Others			保留不用	
[7:4]	-	保留不用				

省电控制寄存器- PRR

PRR – 省电控制寄存器								
PRR: 0x64					默认值: 0x00			
PRR	PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUART0	PRADC
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
位定义								
[0]	PRADC	设置为 1，关闭 ADC 控制器时钟						
[1]	PRUART0	设置为 1，关闭 USART0 模块的时钟						
[2]	PRSPI	设置为 1，关闭 SPI 模块的时钟						
[3]	PRTIM1	设置为 1，关闭定时/计数器 1 的时钟						
-	-	保留不用						
[5]	PRTIM0	设置为 1，关闭定时/计数器 0 的时钟						
[6]	PRTIM2	设置为 1，关闭定时/计数器 2 的时钟						
[7]	PRTWI	设置为 1，关闭 TWI 模块的时钟						

省电控制寄存器- PRR1

PRR1 – 省电控制寄存器 1								
PRR1: 0x65					默认值: 0x00			
PRR1			PRWDT	-	PRTIM3	PREFL	PRPCI	-
R/W			R/W	-	R/W	R/W	R/W	-
位定义								
[0]	-	保留不用						
[1]	PRPCI	设置为 1，关闭外部引脚变化以及外部中断模块时钟						
[2]	PREFL	设置为 1，关闭 FLASH 控制器接口时钟						
[3]	PRTIM3	设置为 1，关闭 TMR3 控制器的时钟						
[4]	-	保留不用						
[5]	PRWDT	设置为 1，关闭 WDT 计数器时钟						
[7:6]	-	保留不用						

MCU 控制寄存器- MCUCR

MCUCR – MCU 控制寄存器								
MCUCR: 0x35(0x55)				默认值: 0x00				
MCUCR	FWKEN	FPDEN	EXRFD	PUD	IRLD	IFAIL	IVSEL	WCE
R/W	R/W	R/W	R/W	R/W	W/O	R/O	R/W	R/W
位定义								
[0]	WCE	MCUCR 更新使能位，在更新 MCUCR 之前，需要首先设置此位，然后在 6 个周期内完成对 MCUCR 寄存器的更新						
[1]	IVSEL	中断向量选择位，此位置 1 后，中断向量地址将根据 IVBASE 寄存器的值映射到新的地址						
[2]	IFAIL	系统配置位加载失败标志位， 0 = 配置信息校验通过 1 = 配置信息加载失败						
[3]	IRLD	写 1 将重新加载系统配置信息						
[4]	PUD	全局上拉禁止位 0 = 使能全局上拉控制 1 = 关闭所有 IO 的上拉电阻						
[5]	EXRFD	外部复位滤波禁止位 0 = 使能外部复位的(190us)数字滤波器 1 = 禁用外部复位的数字滤波电路						
[6]	FPDEN	Flash Power/down 使能控制 0：系统 SLEEP 后 FLASH 保持上电状态 1：系统 SLEEP 后 FLASH 断电						
[7]	FWKEN	快速唤醒模式使能控制，仅对 Power/Off 模式有效 0：260us 滤波延时 1：32us 滤波延时						

PD 组电平变化唤醒控制寄存器- IOCWK

IOCWK – PD 组电平变化唤醒控制寄存器								
IOCWK: 0xAE					默认值: 0x00			
Bits	IOCD7	IOCD6	IOCD5	IOCD4	IOCD3	IOCD2	IOCD1	IOCD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	IOCWK	置 1 对应的位，使能 PD 组 IO 的引脚电平变化唤醒功能						

DPS2 模式控制寄存器- DPS2R

DPS2R – DPS2 模式控制寄存器								
DPS2R: 0xAF					默认值: 0x00			
Bits	-	-	-	-	DPS2E	LPRCE	TOS1	TOS0
R/W	-	-	-	-	R/W	R/W	R/W	R/W
位定义								
[1:0]	TOS	LPRC 定时唤醒设置： 00 = 128ms 01 = 256ms 10 = 512ms 11 = 1s						
[2]	LPRCE	LPRC 使能控制 0 = 禁用 LPRC 定时器 1 = 使能 LPRC 定时器						
[3]	DPS2E	DPS2 模式使能控制位 0 = 禁用 DPS2 模式 1 = 使能 DPS2 模式						
[7:4]	-	保留						

系统控制与复位

概述

系统复位以后，所有的 I/O 寄存器都会被设置为它们的初始值，程序从复位向量处开始执行。LGT8FX8P 的中断向量地址上， 必须用一个 RJMP – 相对跳转指令跳转到复位处理程序。如果程序没用使用到中断，没有使能中断源，中断向量也就不会被使用，中断向量区域就可以用来存放用户的程序代码。

复位有效后，所有 I/O 端口立即进入它们的初始状态。大部分 I/O 的初始化状态为输入并关闭掉内部上拉电阻。有模拟输入功能的 I/O，也初始化为数字 I/O 功能。

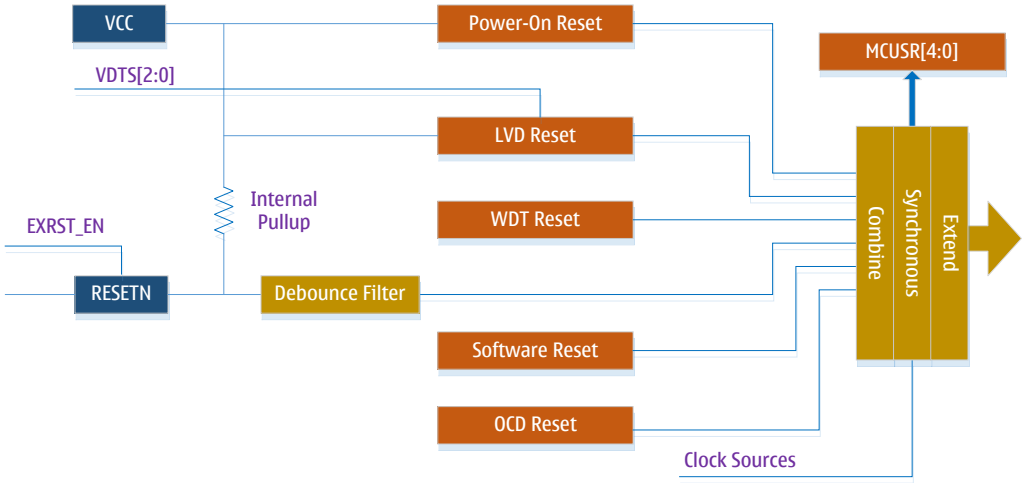
当复位变为无效后，LGT8FX8P 内部的定时计数器开始启动，用于展宽复位。展宽复位信号的宽度用于保证系统中的电源以及时钟等模块进入到稳定的状态。

复位源

LGT8FX8P 共支持六种复位源：

- 上电复位：当系统的工作电压低压内部 POR 模块的复位阀值时，上电复位有效。
- 外部复位：在芯片的外部复位引脚上一定宽度的低电平脉冲，外部复位有效。
- 看门狗复位：使能看门狗模块后，如果看门狗定时器超时，系统将会复位。
- 低电压复位：LGT8FX8P 内部有一个低电压检测模块(LVD)，当系统工作电源低于 LVD 设定的复位阀值时，MCU 也将会被复位。
- 软件复位: LGT8FX8P 内部有一个专用的软件触发的复位寄存器，用户可以通过这个寄存器随时复位 MCU。
- OCD 复位：OCD 复位是有调试器模块发出的，用于直接复位 MCU 内核。

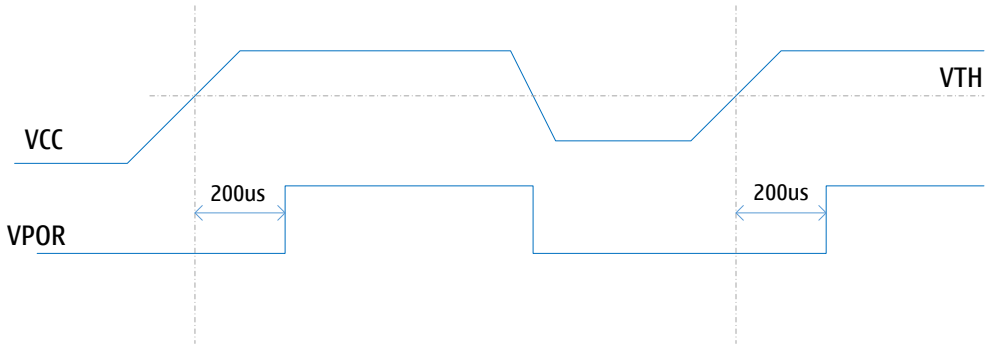
复位系统结构图：



上电复位

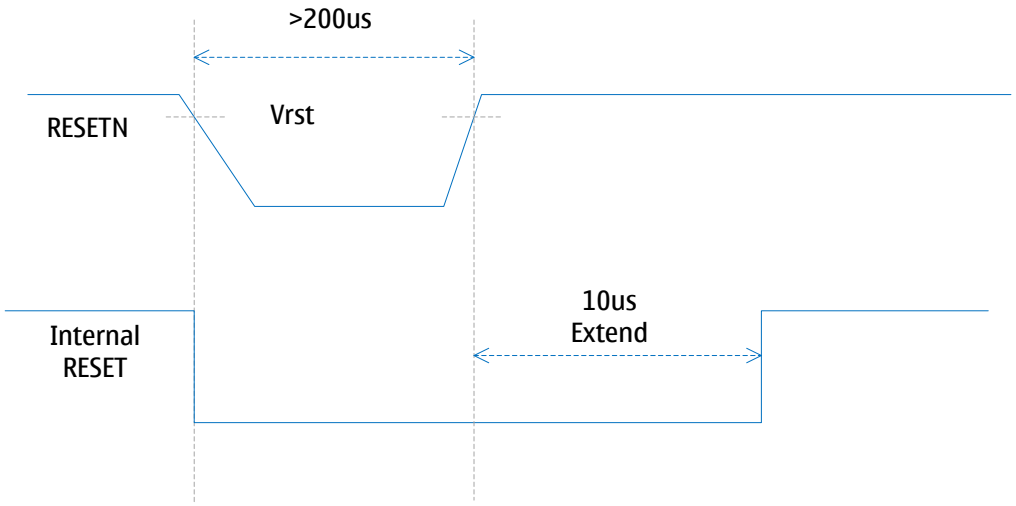
上电复位信号由内部的电压检测电路产生。当系统电源(VCC)低于检测阀值时，上电复位信号有效。上电复位的检测阀值，请参考电气参数部分。

上电复位电路能够保证芯片在上电过程中处于复位状态，芯片上电后能够从一个已知的稳定的状态开始运行。上电复位信号也会被芯片内部的计数器展宽，以保证上电后内部的各种模拟模块，比如 RC 振荡器等能够进入稳定的工作状态。



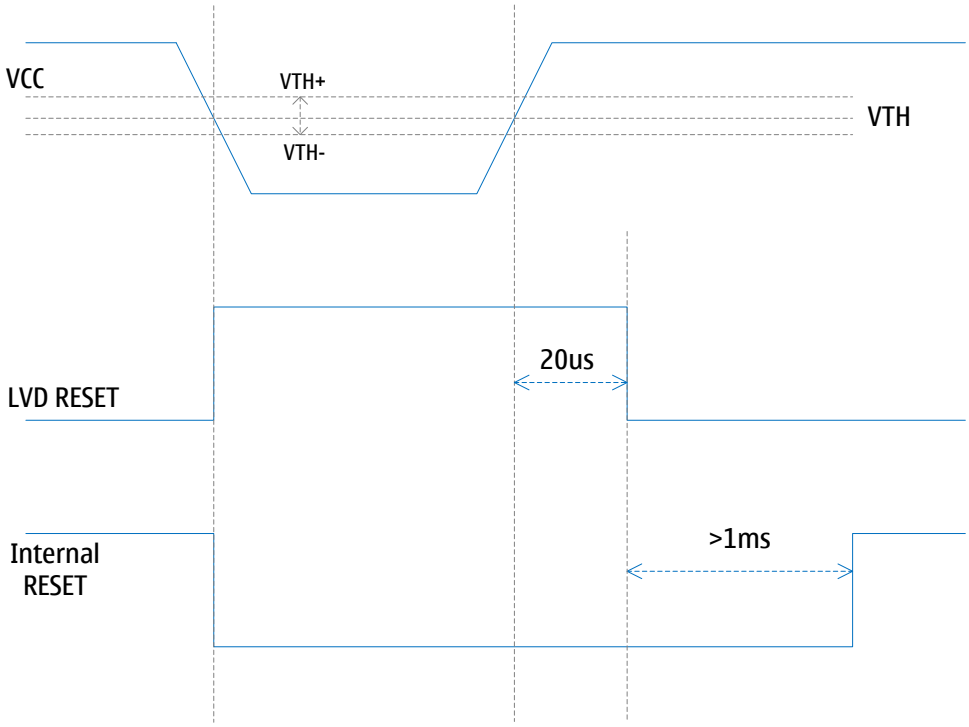
外部复位

在外部复位引脚(RSTN)上施加一个低电平，外部复位立即有效。低电平的宽度要大于一个最小复位脉冲宽度要求。外部复位为异步复位，即使芯片没有时钟工作，外部复位仍然能够对芯片进行复位。LGT8FX8P 的外部复位引脚同时也可以作为通用 I/O 使用。在芯片上电以后， 默认作为外部复位功能。用户可以通过寄存器配置，关闭该引脚的外部复位功能，从而可以当作普通的 I/O 使用。具体使用请参考 IOCR 寄存器的描述部分。



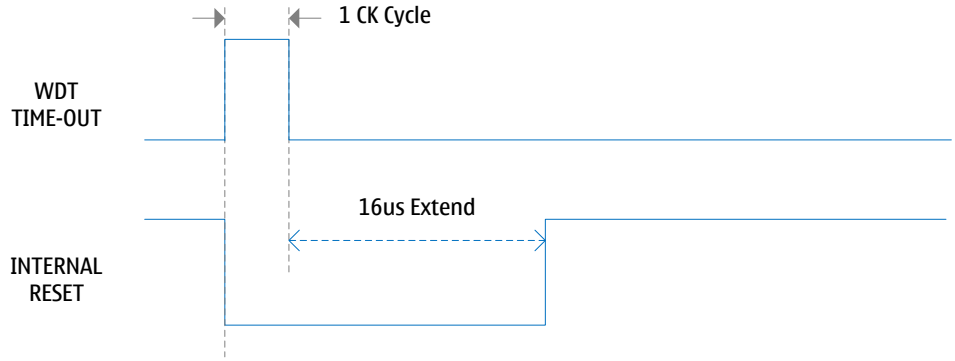
低电压检测(LVD)复位

LGT8FX8P 内部包含一个可编程低电压检测(LVD)电路。LVD 同样是检测 VCC 的电压变化，但与上电复位不同的是，LVD 可以选择检测电压的阀值。用户可以通过直接通过操作 VDTCR 寄存器在不同的电压阀值之间选择。LVD 的电压检测电路具有 ±10mV~±50mV 的迟滞特性，用于滤除 VCC 电压的抖动。当 LVD 使能后，如果 VCC 的电压下降到设定的复位阀值，LVD 复位将立刻有效。当 VCC 增加到复位阀值以上后，内部的复位展开电路启动，将复位继续展宽至少 1 毫秒。



看门狗复位

当看门狗定时器溢出时，如果使能了看门狗系统复位功能，将立刻产生一个周期的系统复位信号。看门狗复位信号通用也会被内部的延时计数器展宽。看门狗控制器的详细操作，请参 考下面的详细介绍部分。



软件复位、OCD 复位

软件复位是用户通过操作 VDTCR 寄存器的第六位触发，软件复位的时序与看门狗复位完全相似。内部将复位信号展宽 16us。

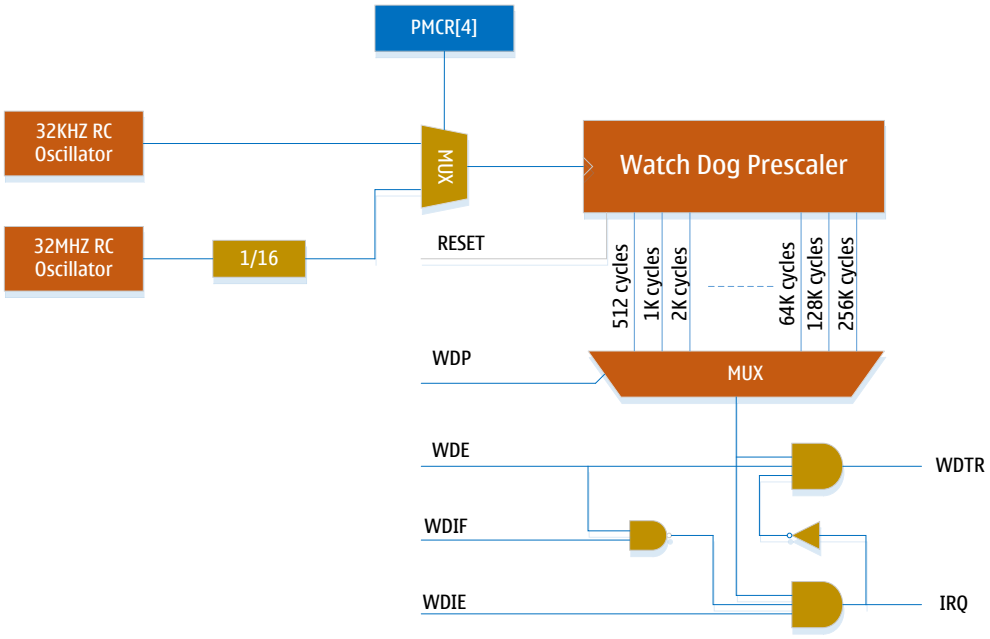
OCD 复位由芯片内部的调试器单元产生，OCD 复位一般是由调试器控制，用户软件无法触发 OCD 复位。

看门狗定时器

- 时钟可选内部 32KHz RC 或内部 32MHz RC 的 16 分频(2MHz)
- 支持中断模式，复位模式以及复位中断模式
- 定时器超时最大可到 8 秒

LGT8FX8P 内部包含一个增强的看门狗定时器(WDT)模块。WDT 定时器的工作时钟可以是内部的 32KHz RC 振荡器，也可以是内部 32MHz RC 振荡器的 16 分频。WDT 计数器溢出后，可以输出一个中断或者一个系统复位信号。在正常使用时，需要软件执行一个 WDR –看门狗定时器复位指令在溢出之前重启计数器。如果系统没有即使的执行 WDR 指令，WDT 将会产生中断或系统复位。

看门狗定时器的结构图如下图所示：



在中断模式下，WDT 溢出后会产生一个中断请求信号。可以使用这个中断作为休眠模式的唤醒信号，也可以作为一个一般的系统定时器使用。比如可以使用这个中断限制某个操作的执行时间，在溢出中终止当前某一个任务。在系统复位模式下，WDT 在计数器溢出后立刻产生一个系统复位信号。最典型的用途就是用于防止系统死机或跑飞。第三种模式，就是复位中断模式，结合了中断和复位两种功能。首先系统将响应 WDT 中断功能，退出 WDT 中断复位程序后，立刻切换到复位模式。这个功能可以支持在复位之前保存一些比较关键的参数信息。

为了防止 WDT 被意外禁止，关闭 WDT 的操作必须按照一个严格定义的时序进行。下面的代码描述如何关闭看门狗定时器。下面的例子假设中断已经被禁止，这样整个操作流程就不会被中断。

看门狗使能以及关闭操作的示例代码：

汇编代码

```
WDT_OFF:
  ; Turn off global interrupt
  CLI
  ; Reset watchdog timer
  WDR
  ; Clear WDRF in MCUSR
  IN r16, MCUSR
  ANDI r16, ~(1 << WDRF)
  OUT MCUSR, r16
  ; Write logical one to WDCE and WDE
  ; Keep old Prescaler setting to prevent unintentional time-out
  LDS r16, WDTCSR
  ORI r16, (1 << WDCE) | (1 << WDE)
  STS WDTCSR, r16
  ; Turn off WDT
  LDI r16, (0 << WDE)
  STS WDTCSR, r16
  ; Turn on global interrupt
  SEI
  RET
```

C 语言代码

```
void WDT_OFF(void)
{
  __disable_interrupt();
  __watchdog_reset();
  /* Clear WDRF in MCUSR */
  MCUSR &= ~(1 << WDRF);
  /* Write logical one to WDCE and WDE */
  /* Keep old Prescaler setting to prevent unintentional time-out */
  WDTCSR |= (1 << WDCE) | (1 << WDE);
  /* Turn off WDT */
  WDTCSR = 0x00;
  __enable_interrupt();
}
```

[使用提示]

如果 WDT 被意外使能，比如程序跑飞，芯片会被复位，但是 WDT 仍然还是在使能状态。如果用户代码里没有处理 WDT，这将会导致循环复位。为避免这种情况，建议用户软件在初始化程序中清除看门狗复位标记位(WDRF)和 WDE 控制位。

下面的代码描述如何改变看门狗定时器的超时值。

汇编代码

```
WDT_TOV_Change:
   ; Turn off global interrupt
   CLI
   ; Reset watchdog timer
   WDR
   ; Start timed sequence
   LDS r16, WDTCSR
   ORI r16, (1 << WDCE) | (1 << WDE)
   STS WDTCSR, r16
   ; -- Got for cycles to set the new value from here --
   ; Set new time-out value = 64k cycles
   LDI r16, (1 << WDE) | (1 << WDP2) | (1 << WDP0)
   STS WDTCSR, r16
   ; -- Finished setting new value, used 2 cycles -
   ; Turn on global interrupt
   SEI
   RET
```

C 语言代码

```
void WDT_TOV_Change(void)
{
   __disable_interrupt();
   __watchdog_reset();
   /* Start timed sequence */
   WDTCSR |= (1 << WDCE) | (1 << WDE);
   /* Set new time-out value = 64K cycles */
   WDTCSR |= (1 << WDE) | (1 << WDP2) | (1 << WDP0);
   __enable_interrupt();
}
```

【使用说明】

在改变 WDP 配置位之前，建议复位看门狗定时器。因为更改 WDP 位到比较小的超时周期很可能会导致看门狗超时复位。

寄存器定义

低压检测(LVD)控制寄存器- VDTCR

VDTCR – LVD 控制寄存器								
VDTCR: 0x62				默认值: 0x00				
Bits	WCE	SWR	-	VDTS2	VDTS1	VDTS0	VDREN	VDTEN
R/W	R/W	W/R	-	R/W	R/W	R/W	R/W	R/W
位定义								

[0]	VDTEN	低压检测模块使能控制，1 使能，0 禁止
[1]	VDREN	低压复位功能使能控制，1 使能，0 禁止
[4:2]	VDTS	低压检测阀值配置位 000 = 1.8V 001 = 2.2V 010 = 2.5V 011 = 2.9V 100 = 3.2V 101 = 3.6V 110 = 4.0V 111 = 4.4V
[5]	-	保留不用
[6]	SWR	软复位使能位，此位清零将产生软件复位
[7]	WCE	VDTCR 值改变使能位 用户在改变 VDTCR 寄存器的值之前，必须首先将此位写 1，在之后的 6 个时钟周期内，更改 VDTCR 其他位的值。四个周期后 WCE 自动清零，对 VDTCR 寄存器的更新操作无效。

IO 功能复用寄存器- PMX2

PMX2 – IO 功能复用寄存器								
PMX2: 0xF0				默认值: 0x00				
Bits	WCE	STSC1	STSC0	-	-	XIEN	E6EN	C6EN
R/W	R/W	R/W	R/W	-	-	R/W	R/W	R/W
位定义								

0	C6EN	PC6 引脚默认为复位功能，设置此位为 1 将禁止外部复位功能，复位功能禁止后，PC6 可作为一个普通的 I/O 使用
1	E6EN	PE6 引脚默认为模拟输入功能，设置此位为 1，将关闭模拟输入功能，这个引脚可以作为 GPIO 使用
2	XIEN	外部时钟输入使能控制
4:3	-	保留不用
5	STSC0	低速晶振启动控制
6	STSC1	高速晶振启动控制
7	WCE	IOCR 值改变使能位 用户在改变 IOCR 寄存器的值之前，必须首先将此位写 1，在

		之后的 6 个时钟周期内，更改 IOCR 其他位的值。四个周期后 WCE 自动清零，对 IOCR 寄存器的更新操作无效。

MCU 状态寄存器- MCUSR

MCUSR – IO 特殊功能控制寄存器								
MCUSR: 0x34(0x54)				默认值: 0x00				
Bits	SWDD	-	PDRF	OCDRF	WDRF	BORF	EXTRF	PORF
R/W	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[0]	PORF	上电复位标志，写 0 清零						
[1]	EXTRF	外部复位标志，上电复位自动清零，或写 0 清零						
[2]	BORF	低电压检测复位，上电复位自动清零，或写 0 清零						
[3]	WDRF	看门狗复位标志，上电复位自动清零，或写 0 清零						
[4]	OCDRF	OCD 调试器复位标志，上电复位自动清零，或写 0 清零						
[5]	PDRF	从 Power/off 模式唤醒标志，具体描述请参考功耗管理章节。						
[6]	-	保留不用						
[7]	SWDD	SWD 接口禁止位。写 1 将关闭 SWD 接口。 SWD 接口关闭后，将无法进行调试和 ISP 操作。如果用户程序中关闭了 SWD 接口，可以通过上电过程中拉低 RESET 的方式禁止内部程序的运行，然后进行调试和 ISP 操作。SWD 接口关闭后，SWD 占用的两个 I/O 接口可以作为通用 I/O 使用。为避免对 SWDD 的误操作，用户需要在第一次更新 SWDD 位之后的四个周期内再写一次 SWDD 才能生效。						

[使用提示]:

为了更加准确有效的使用复位标志信息，建议用户尽量在程序的初始化前期读取复位标志然后将其清零。

看门狗控制状态寄存器- WDTCSR

WDTCSR – WDT 控制和状态寄存器								
地址: 0x60					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	WDIF	WDIE	WDP3	WDTOE	WDE	WDP2	WDP1	WDP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
[7]	WDIF	WDT 中断标志位。 当 WDT 工作在中断模式并发生溢出时会置位 WDIF 位。当 WDT 中断使能位 WDIE 为“1”且全局中断置位时，WDT 中断产生。执行 WDT 中断时会清零 WDIF 位，对 WDIF 位写“1”也可清零该位。
[6]	WDIE	WDT 中断使能控制位。 当设置 WDIE 位为“1”，且全局中断置位时，WDT 中断被使能。

<table>
<tr><td></td><td></td><td>当设置 WDIE 位为“0”时，WDT 中断被禁止。
WDIE 位和 WDE 位一起决定看门狗的工作模式，如下表所示。
<table>
<tr><th>WDE</th><th>WDIE</th><th>模式</th><th>溢出后动作</th></tr>
<tr><td>0</td><td>0</td><td>停止</td><td>无</td></tr>
<tr><td>0</td><td>1</td><td>中断模式</td><td>中断</td></tr>
<tr><td>1</td><td>0</td><td>复位模式</td><td>复位</td></tr>
<tr><td>1</td><td>1</td><td>中断复位模式</td><td>中断后复位</td></tr>
</table></td></tr>
<tr><td>[5]</td><td>WDP3</td><td>WDT 预分频因子选择控制第 3 位。
WDP[3]和 WDP[2:0]组成 WDT 预分频因子选择位 WDP[3:0]，用来设置 WDT 的溢出周期。</td></tr>
<tr><td>[4]</td><td>WDTOE</td><td>WDT 关闭使能控制位。
当要把 WDE 位清零时，WDTOE 位须置位，否则 WDT 不会被关闭。
当 WDTOE 位被置位后，硬件会在 4 个时钟周期后清零 WDTOE 位。</td></tr>
<tr><td>[3]</td><td>WDE</td><td>WDT 使能控制位。
当设置 WDE 位为“1”时，WDT 被使能。当设置 WDE 位为“0”时，WDT 被禁止。
只有在 WDTOE 位置位时 WDE 才能被清零。要关闭已经使能了的 WDT，必须按照下列时序操作：
1. 同时置位 WDTOE 和 WDE 位，即使 WDE 已经被置位，在关闭操作开始之前也必须对 WDE 位写入“1”；
2. 在接下来的 4 个时钟周期内，对 WDE 位写入“0”。这将关闭 WDT。
当 WDE 位为“1”且 WDT 溢出复位系统后会置位 WDT 复位系统标志 WDRF（位于 MCUSR 寄存器）。当 WDRF 位处于置位状态时会置位 WDE 位。因此要清零 WDE 位，必须先清零 WDRF 位。</td></tr>
<tr><td>[2:0]</td><td>WDP</td><td>WDT 预分频因子选择控制。
用来设置 WDT 的溢出周期。建议在 WDT 未计数时改变 WDP 的值，在计数过程中改变 WDP 的值就会产生不可预期的 WDT 溢出。</td></tr>
</table>

看门狗预分频选择列表：

WDP3	WDP2	WDP1	WDP0	看门狗定时器 溢出周期数	32KHz 时钟	2MHz 时钟
0	0	0	0	2K cycles	64ms	1ms
0	0	0	1	4K cycles	128ms	2ms
0	0	1	0	8K cycles	256ms	4ms
0	0	1	1	16K cycles	512ms	8ms
0	1	0	0	32K cycles	1s	16ms
0	1	0	1	64K cycles	2s	32ms
0	1	1	0	128K cycles	4s	64ms
0	1	1	1	256K cycles	8s	128ms
1	0	0	0	512K cycles	16s	256ms
1	0	0	1	1024K cycles	32s	512ms

1	0	1	0	保留不用
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

中断与中断向量

- 28 个中断源
- 可编程向量起始地址

LGT8F88P/168P/328P 的中断资源基本相同，主要的区别为：LGT8F88P 的中断向量为 1 个指令字(16 位)，而 LGT8F168P/328P 的中断向量为 2 个指令字。

LGT8F88P 中断向量列表

LGT8F88P 中断向量列表：

编号	向量地址	中断源信号	中断源说明
1	0x0000	RESET	外部复位，上电复位，看门狗复位，SWD 调试复位，低电压复位
2	0x0001	INT0	外部中断请求 0
3	0x0002	INT1	外部中断请求 1
4	0x0003	PCI0	引脚电平中断 0
5	0x0004	PCI1	引脚电平中断 1
6	0x0005	PCI2	引脚电平中断 2
7	0x0006	WDT	看门狗溢出中断
8	0x0007	TC2 COMPA	定时器 2 比较匹配 A 中断
9	0x0008	TC2 COMPB	定时器 2 比较匹配 B 中断
10	0x0009	TC2 OVF	定时器 2 溢出中断
11	0x000A	TC1 CAPT	定时器 1 输入捕捉中断
12	0x000B	TC1 COMPA	定时器 1 比较匹配 A 中断
13	0x000C	TC1 COMPB	定时器 1 比较匹配 B 中断
14	0x000D	TC1 OVF	定时器 1 溢出中断
15	0x000E	TC0 COMPA	定时器 0 比较匹配 A 中断
16	0x000F	TC0 COMPB	定时器 0 比较匹配 B 中断
17	0x0010	TC0 OVF	定时器 0 溢出中断
18	0x0011	SPI STC	SPI 串行传输结束中断
19	0x0012	USART RXC	USART 接收结束中断
20	0x0013	USART UDRE	USART 数据寄存器空中断
21	0x0014	USART TXC	USART 发送结束中断
22	0x0015	ADC	ADC 转换结束中断
23	0x0016	EE_RDY	EEPROM 就绪中断
24	0x0017	ANA_COMP	模拟比较器 0 中断
25	0x0018	TWI	两线串行接口中断
26	0x0019	ANA_COMP1	模拟比较器 1 中断
27	0x001A	-	保留
28	0x001B	PCI3	引脚电平中断 3
29	0x001C	PCI4	引脚电平中断 4
30	0x001D	TC3_INT	定时器 3 中断

LGT8F168P/328P 中断向量列表

LGT8F168P/328P 中断向量列表：

编号	向量地址	中断源信号	中断源说明
1	0x0000	RESET	外部复位，上电复位，看门狗复位，SWD 调试复位，低电压复位
2	0x0002	INT0	外部中断请求 0
3	0x0004	INT1	外部中断请求 1
4	0x0006	PCI0	引脚电平中断 0
5	0x0008	PCI1	引脚电平中断 1
6	0x000A	PCI2	引脚电平中断 2
7	0x000C	WDT	看门狗溢出中断
8	0x000E	TC2 COMPA	定时器 2 比较匹配 A 中断
9	0x0010	TC2 COMPB	定时器 2 比较匹配 B 中断
10	0x0012	TC2 OVF	定时器 2 溢出中断
11	0x0014	TC1 CAPT	定时器 1 输入捕捉中断
12	0x0016	TC1 COMPA	定时器 1 比较匹配 A 中断
13	0x0018	TC1 COMPB	定时器 1 比较匹配 B 中断
14	0x001A	TC1 OVF	定时器 1 溢出中断
15	0x001C	TC0 COMPA	定时器 0 比较匹配 A 中断
16	0x001E	TC0 COMPB	定时器 0 比较匹配 B 中断
17	0x0020	TC0 OVF	定时器 0 溢出中断
18	0x0022	SPI STC	SPI 串行传输结束中断
19	0x0024	USART RXC	USART 接收结束中断
20	0x0026	USART UDRE	USART 数据寄存器空中断
21	0x0028	USART TXC	USART 发送结束中断
22	0x002A	ADC	ADC 转换结束中断
23	0x002C	EE_RDY	EEPROM 就绪中断
24	0x002E	ANA_COMP	模拟比较器中断
25	0x0030	TWI	两线串行接口中断
26	0x0032	ANA_COMP1	模拟比较器 1 中断
27	0x0034	-	保留
28	0x0036	PCI3	引脚电平中断 3
29	0x0038	PCI4	引脚电平中断 4
30	0x003A	TC3_INT	定时器 3 中断

LGT8FX8P 的复位向量从地址 0x0000 开始执行。除复位向量外，其他向量地址都可以通过 MCUCR 寄存器中的 IVSEL 以及 IVBASE 寄存器重新定向到 512 字节对齐的起始地址。

中断向量处理

下面代码仅以 LGT8F88P 为例，用于说明复位以及中断向量编程，仅供参考：

汇编代码实例 – LGT8F88P		
地址	代码	说明
0x000	RJMP RESET	复位向量
0x001	RJMP EXT_INT0	外部中断 0
0x002	RJMP EXT_INT1	外部中断 1
0x003	RJMP PCINT0	引脚电平变化中断 0
0x004	RJMP PCINT1	引脚电平变化中断 1
0x005	RJMP PCINT2	引脚电平变化中断 2
0x006	RJMP WDT	看门狗定时器中断
0x007	RJMP TIM2_COMPA	定时器 2 比较匹配 A 组中断
0x008	RJMP TIM2_COMPB	定时器 2 比较匹配 B 组中断
0x009	RJMP TIM2_OVF	定时器 2 溢出中断
0x00A	RJMP TIM1_CAPT	定时器 1 俘获中断
0x00B	RJMP TIM1_COMPA	定时器 1 比较匹配 A 组中断
0x00C	RJMP TIM1_COMPB	定时器 1 比较匹配 B 组中断
0x00D	RJMP TIM1_OVFR	定时器 1 溢出中断
0x00E	RJMP TIM0_COMPA	定时器 0 比较匹配 A 组中断
0x00F	RJMP TIM0_COMPB	定时器 0 比较匹配 B 组中断
0x010	RJMP TIM0_OVF	定时器 0 溢出中断
0x011	RJMP SPI_STC	SPI 传输完成中断
0x012	RJMP USART_RXC	USART 接收完成中断
0x013	RJMP USART_UDRE	USART 数据寄存器空中断
0x014	RJMP USART_TXC	USART 发送完成中断
0x015	RJMP ADC	ADC 转换完成中断
0x016	RJMP EE_RDY	EEPROM 控制器准备好中断
0x017	RJMP ANA_COMP	比较器中断
0x018	RJMP TWI	TWI 控制器中断
0x019	NOP	保留地址
0x01A	NOP	保留地址
0x01B	RJMP PCI3	引脚电平变化中断 3
;		
0x01C (**RESET :**)	LDI r16, high(RAMEND)	主程序开始
0x01D	OUT SPH, r16	设置堆栈指针为 RAM 顶端地址
0x01E	LDI r16, low(RAMEND)	
0x01F	OUT SPL, r16	
0x020	SEI	使能全局中断
0x021	.....	

寄存器定义

MCU 控制寄存器- MCUCR

MCUCR – MCU 控制寄存器								
MCUCR: 0x35(0x55)				默认值: 0x00				
MCUCR	FWKEN	FPDEN	EXRFD	PUD	IRLD	IFAIL	IVSEL	WCE
R/W	R/W	R/W	R/W	R/W	W/O	R/O	R/W	R/W
位定义								
[0]	WCE	MCUCR 更新使能位，在更新 MCUCR 之前，需要首先设置此位，然后在 6 个周期内完成对 MCUCR 寄存器的更新						
[1]	IVSEL	中断向量选择位，此位置 1 后，中断向量地址将根据 IVBASE 寄存器的值映射到新的地址						
[2]	IFAIL	系统配置位加载失败标志位， 0 = 配置信息校验通过 1 = 配置信息加载失败						
[3]	IRLD	写 1 将重新加载系统配置信息						
[4]	PUD	全局上拉禁止位 0 = 使能全局上拉控制 1 = 关闭所有 IO 的上拉电阻						
[5]	EXRFD	外部复位滤波禁止位 0 = 使能外部复位的(190us)数字滤波器 1 = 禁用外部复位的数字滤波电路						
[6]	FPDEN	Flash Power/down 使能控制 0：系统 SLEEP 后 FLASH 保持上电状态 1：系统 SLEEP 后 FLASH 断电						
[7]	FWKEN	快速唤醒模式使能控制，仅对 Power/Off 模式有效 0：260us 滤波延时 1：32us 滤波延时						

中断向量基地址寄存器 - IVBASE

IVBASE – 中断向量基地址寄存器		
IVBASE: 0x75	默认值: 0x00	
IVBASE	IVBASE[7:0]	
R/W	R/W	
位定义		
[7:0]	IVBASE	如果 IVSEL 为 1，中断向量(复位向量除外)将以 IVBASE 为基地址在 512 字节的页面上重新映射。 映射后的中断向量基地址为: (IVBASE << 8) + 表 1 中对应的向量地址

外部中断

- 2 个外部中断源
- 可配置的电平或边沿触发中断
- 可用作睡眠模式下的唤醒源

概述

外部中断由 INT0 和 INT1 引脚触发。只要外部中断被使能，即使这 2 个引脚配置为输出也能触发中断。这可以用来产生软件中断。外部中断可以由上升沿，下降沿或低电平触发，由外部中断控制寄存器 EICRA 来配置。当外部中断使能并且配置为电平触发（只有 INT0 和 INT1 引脚）时，只要引脚电平为低，中断就会一直产生。INT0 和 INT1 引脚的上升沿或下降沿中断触发需要 IO 时钟正常工作，而 INT0 和 INT1 引脚的低电平触发中断都是异步检测的。除了空闲模式，其它睡眠模式下 IO 时钟都是停止工作的。因此，这 2 个外部中断都可用作除空闲模式外的其它睡眠模式下的唤醒源。

若电平触发中断用作省电模式下的唤醒源，改变的电平必须保持一定的时间来唤醒 MCU，以降低 MCU 对噪声的敏感程度。要求的电平必须保持足够长的时间使 MCU 结束唤醒过程，然后触发电平中断。

寄存器定义

寄存器列表

寄存器	地址	默认值	描述
EICRA	0x69	0x00	外部中断控制寄存器 A
EIMSK	0x3D	0x00	外部中断屏蔽寄存器
EIFR	0x3C	0x00	外部中断标志寄存器

外部中断控制寄存器 A- EICRA

EICRA – 外部中断控制寄存器 A								
地址: 0x69				默认值: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	ISC11	ISC10	ISC01	ISC00
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	描述
7:4	-	保留。
3	ISC11	INT1 引脚中断触发方式控制位高位。
2	ISC10	INT1 引脚中断触发方式控制位低位。 当全局中断置位且 GICR 寄存器的相应中断屏蔽控制位被置位时，外部中断 1 由 INT1 引脚激发。中断的触发方式见表格描述。在边沿检测之前 MCU 首先采样 INT1 引脚上的电平。如果选用了边沿触发方式或电平变化触发方式，那么持续时间大于 1 个系统时钟周期的脉冲将触发中断，过短的脉冲则不能保证触发中断。如果选择低电

		平触发方式，那么低电平必须保持到当前指令执行完成才会触发中断。
1	ISC01	INT0 引脚中断触发方式控制位高位。
0	ISC00	INT0 引脚中断触发方式控制位低位。 当全局中断置位且 GICR 寄存器的相应中断屏蔽控制位被置位时，外部中断 0 由 INT0 引脚激发。中断的触发方式见表格描述。在边沿检测之前 MCU 首先采样 INT0 引脚上的电平。如果选用了边沿触发方式或电平变化触发方式，那么持续时间大于 1 个系统时钟周期的脉冲将触发中断，过短的脉冲则不能保证触发中断。如果选择低电平触发方式，那么低电平必须保持到当前指令执行完成才会触发中断。

外部中断 1 触发方式见下表。

外部中断 1 触发方式控制

ISC1[1:0]	描述
0	外部引脚 INT1 低电平触发
1	外部引脚 INT1 上升沿或下降沿触发
2	外部引脚 INT1 下降沿触发
3	外部引脚 INT1 上升沿触发

外部中断 0 触发方式见下表。

外部中断 0 触发方式控制

ISC0[1:0]	描述
0	外部引脚 INT0 低电平触发
1	外部引脚 INT0 上升沿或下降沿触发
2	外部引脚 INT0 下降沿触发
3	外部引脚 INT0 上升沿触发

外部中断屏蔽寄存器- EIMSK

EIMSK – 外部中断屏蔽寄存器								
地址: 0x3D						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	-	-	INT1	INT0
R/W	-	-	-	-	-	-	R/W	R/W
Bit	Name	描述						
7:2	-	保留						
1	INT1	外部引脚 1 中断使能控制位。 当设置 INT1 位为“1”时，且全局中断置位，外部引脚 1 中断被使能，唤醒功能被使能。即使 INT1 引脚被配置为输出，只要引脚电平发生了相应的变化，中断将产生。 当设置 INT1 位为“0”时，外部引脚 1 中断被禁止，唤醒功能也被禁止。						

0	INT0	外部引脚 0 中断使能控制位。 当设置 INT0 位为“1”时，且全局中断置位，外部引脚 0 中断被使能，唤醒功能被使能。即使 INT0 引脚被配置为输出，只要引脚电平发生了相应的变化，中断将产生。 当设置 INT0 位为“0”时，外部引脚 0 中断被禁止，唤醒功能也被禁止。

外部中断标志寄存器- EIFR

EIFR – 外部中断标志寄存器								
地址: 0x3C					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	-	-	INTF1	INTF0
R/W	-	-	-	-	-	-	R/W	R/W

Bit	Name	描述
7:2	-	保留。
1	INTF1	外部引脚 1 中断标志位。 当边沿触发外部引脚 1 中断时，INTF1 被置位。当低电平触发外部引脚 1 中断时，不会置位 INTF1 位。若此时外部引脚 1 中断使能 INT1EN 位为“1”且全局中断标志置位，则会产生外部引脚 1 中断。执行此中断服务程序时 INTF1 将自动清零，或对 INTF1 位写“1”也可清零该位。
0	INTF0	外部引脚 0 中断标志位。 当边沿触发外部引脚 0 中断时，INTF0 被置位。当低电平触发外部引脚 0 中断时，不会置位 INTF0 位。若此时外部引脚 0 中断使能 INT0EN 位为“1”且全局中断标志置位，则会产生外部引脚 0 中断。执行此中断服务程序时 INTF0 将自动清零，或对 INTF0 位写“1”也可清零该位。

运算加速器(uDSC)

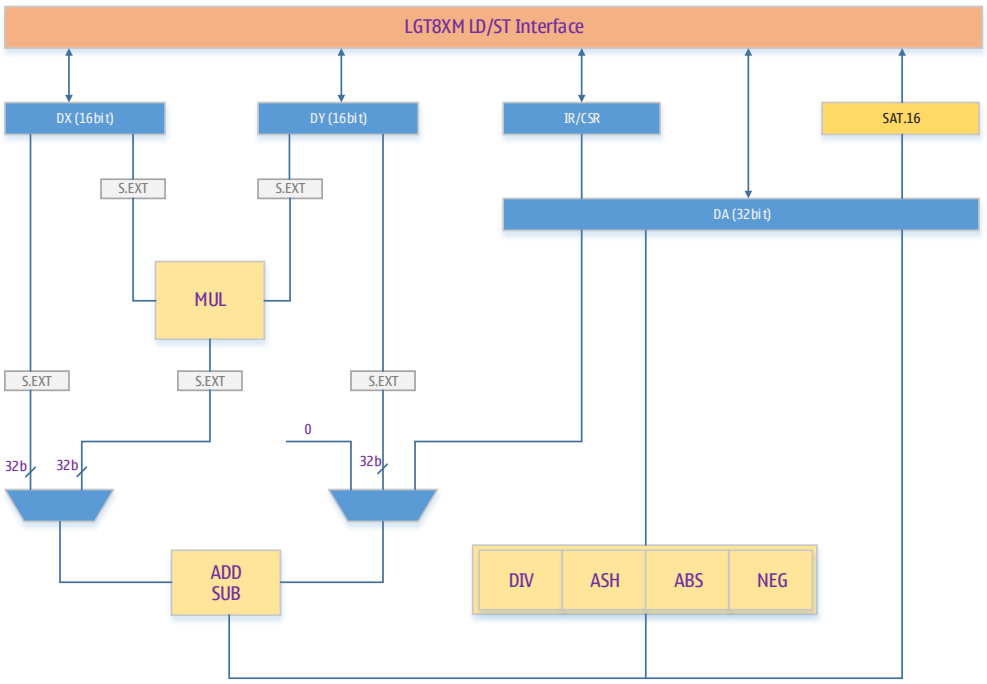
- 16 位存储模式(LD/ST)
- 32 位累加器(DX)
- 单周期 16 位乘法器(MUL)
- 32 位算术逻辑运算单元(ALU)
- 16 位饱和运算(SD)
- 8 周期 32/16 除法器
- 单周期乘加/乘减运算(MAC/MSC)

概述

数字运算加速器（uDSC）作为 LGT8XM 内核的一个运算协处理模块，配合 LGT8XM 内核 16 位 LD/ST 模式，实现一个 16 位的数字信号处理单元。可以满足大部分控制类数字信号的处理。

uDSC 功能内部以及功能：

1. 16 位操作数寄存器 DX/DY
2. 32 位累加寄存器 DA
3. 单周期 17 位乘法器（可以实现 16 位有/无符号乘法运算）
4. 32 位 ALU (可以实现 16/32 位的加法，减法以及移位运算)
5. 16 位饱和运算 （用于将运算结果存储到 RAM 空间）
6. 32/16 除法器，8 个周期内完成运算



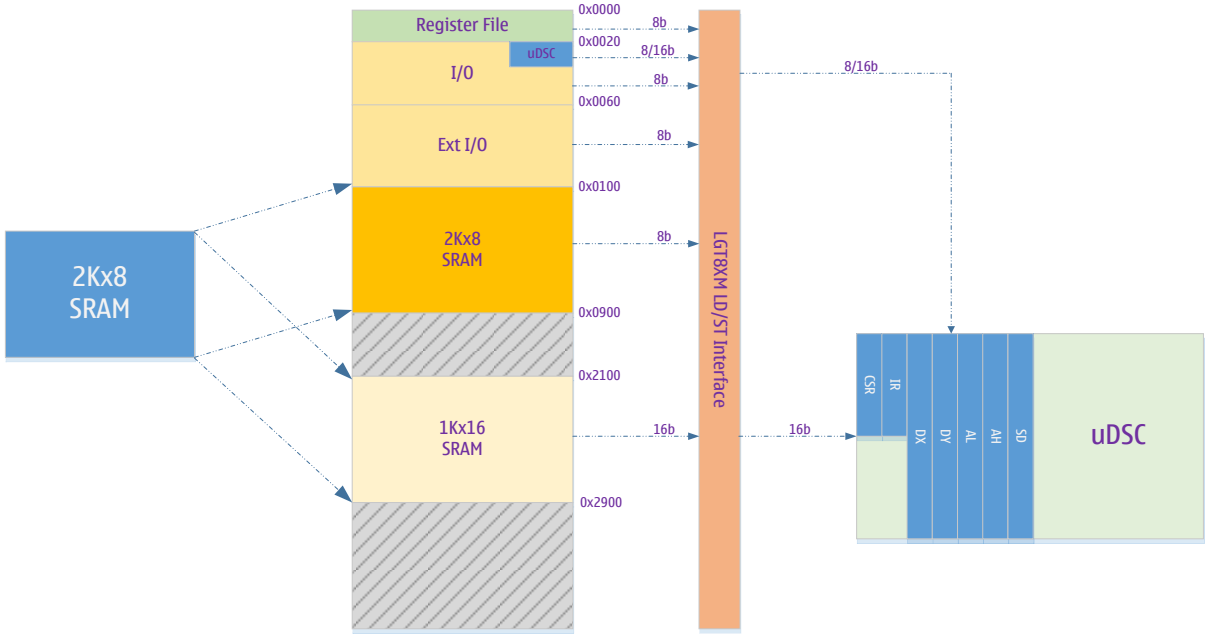
uDSC 结构图

16 位 LD/ST 工作模式

为提高 uDSC 处理大量数据运算的效率，LGT8XM 内核实现一个专用的 16 位 LD/ST 存储通道，可以使用 LDD/STD 指令高效的在 uDSC 与 SRAM 以及通用寄存器文件之间进行 16 位数据交换。

为了不破坏正常的 LD/ST 指令系统，LGT8XM 内核把 SRAM 空间重映射到 0x2100~0x28FF。使用 LD/ST 指令从 0x2100~0x28FF 空间访问 SRAM 时， 内核自动开启 16 位 LD/ST 功能，打开 SRAM 与 uDSC 之间的直接存取通道。

下图为 LGT8XM 内核的数据空间地址分布：



如上图所示，LGT8XM 内核可以通过使用 LD/ST 指令，在 uDSC 的 DX/DY/DA 寄存器与 SRAM 之间直接进行 16 位的数据存存取访问。同时 uDSC 的内部寄存器也映射到 I/O 空间，访问 uDSC 寄存器分为 8/16 两种模式.

uDSC 内部除了用于运算的 DX/DY/DA 寄存器外， 还包含了另外 2 个 8 位的寄存器：uDSC 控制状态寄存器 CSR 以及运算指令寄存器 IR。CSR/IR 只能通过 I/O 空间以字节为单位访问；访问 DX/DY/AL/AH 时为 16 位模式。可以使用 IN/OUT 以及 LD/ST/LDD/STD/LDS/STD 等指令访问。

uDSC 相关的控制状态以及数据寄存器均映射到 IO 空间，直接使用 IN/OU 指令寻址，可以在一个指令周期内完成 8/16 位的数据访问。

CSR 用于控制 uDSC 的工作模式以及记录当前 uDSC 执行运算的状态标志位。IR 控制 uDSC 实现的具体运算。uDSC 支持的运算大部分都会在一个周期内完成，除法运行需要 7 个等待周期，也可以通过 CSR 寄存器中的标志位判断当前的除法操作是否完成。

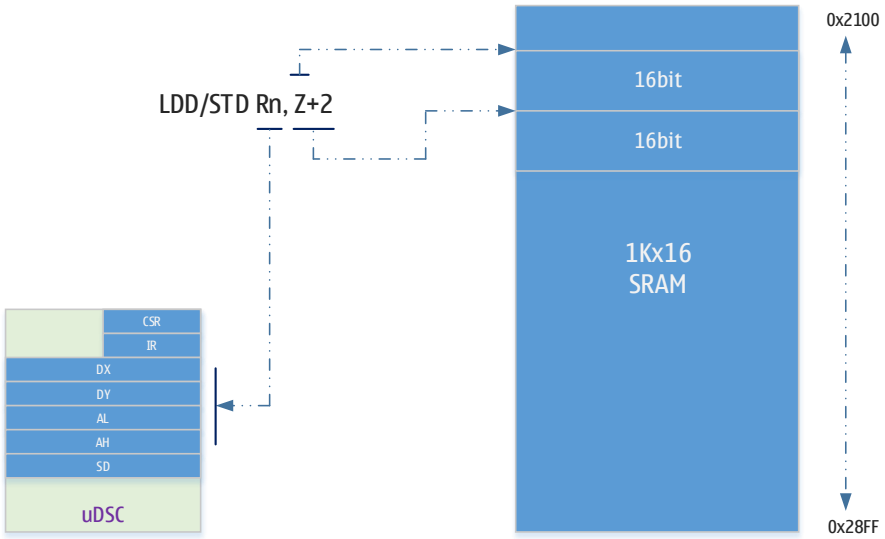
标准的 LD/ST 指令使用 LGT8XM 内部的通用工作寄存器作为 LD/ST 的数据，使用 X/Y/Z 作为目标地址。当目标地址落在 16 位 SRAM 映射空间时，此时 LD/ST 指令操作数的含义有所变化，其中 X/Y/Z 仍然作为目标地址，通用工作寄存器寻址的含义根据 uDSC 映射模式将有两种处理方式。uDSC 的映射模式只作用于对 0x2100~0x28FF 地址存取访问。映射模式通过 CSR 寄存器的第 6 位(MM)设置。

16 位 LD/ST 模式下，指令”LDD Rn, Z+q” 表示的是把[Z]地址的 16 位数据加载到 uDSC 的数据寄存器中，然后将 Z 的值增加一个偏移量”q”。此处 Rn 的含义与映射模式 CSR[MM]的关系如下：

LDD Rn, Z/Y+q			
CSR[MM]	[Z+q]	Opcode	Operations
0	0x2100~0x28FF	LDD R0, Z+q	DX = [Z]; Z = Z + q; R0 kept unchanged
		LDD R1, Z+q	DY = [Z]; Z= Z + q; R1 kept unchanged
		LDD R2, Z+q	AL = [Z]; Z= Z + q; R2 kept unchanged
		LDD R3, Z+q	AH = [Z]; Z= Z + q; R3 kept unchanged
1	0x2100~0x28FF	LDD Rn, Z+q	{Rn} address for DX/DY/AL/AH in I/O region [DX/DY/AL/AY] = [Z]; Z = Z + q Rn keep unchanged
STD Rn, Z/Y+q			
0	0x2100~0x28FF	STD Z+q, R0	[Z] = DX; Z = Z + q; R0 kept unchanged
		STD Z+q, R1	[Z] = DY; Z = Z + q; R1 kept unchanged
		STD Z+q, R2	[Z] = AL; Z = Z + q; R2 kept unchanged
		STD Z+q, R3	[Z] = AH; Z = Z + q; R3 kept unchanged
		STD Z+q, R4	[Z] = SD; Z = Z + q; R4 kept unchanged
1	0x2100~0x28FF	STD Z+q, Rn	{Rn} address for DX/DY/AL/AH/SD in I/O region [Z] = [DX/DY/AL/AH/SD] addressed by {Rn} Rn keep unchanged

LGT8XM 指令集中的 LD/ST, LDS/STS 都可以访问到 0x2100~0x28FF 区域，但是 LDD/STD 的 Y/Z+q 寻址方式更加有效。LDD/STD 方式的寻址基于一个基地址，我们可以把 Y/Z 设置为 RAM 中数据的基地址，通过使用 LDD/STD 指令的 Y/Z+q 寻址方式，可以在一个周期内执行指令和存取数据，并将地址指针自动移动到下一个目标地址。

LGT8XM 内核标准的 LDD/STD 指令的 Y/Z+q 偏移寻址模式，指令执行时使用[Y/Z+q]作为 8 位数据的地址，执行完成后 Y/Z 的值并不增加。当使用 LDD/STD 寻址 0x2100~0x28FF 区间的地址时，LDD/STD 的指令行为发生改变：指令执行时，使用[Y/Z]作为 16 位数据寻址地址，执行后，Y/Z 的值增加”q”指定的偏移量。这种特性可以提高我们连续寻址的效率， 通过将”q=2”可以实现连续 16 位数据的寻址。



变量地址与 16 位模式地址之间的映射

LGT8XM 为 8 位处理器，数据访问以字节为单位。LGT8F328P 内置 2K 字节的数据空间。这部分空间被映射到 0x0100~0x08FF 的地址。C/C++编译自动将变量分配到 0x0100~0x08FF 之间。如果我们在 C/C++中定义的一个 16 位的数组需要使用 uDSC 进行运算，就需要首先将该变量的地址映射到 16 位 LD/ST 访问的地址区域(0x2100~0x28FF)。方法很简单，只需要将变量的地址增加 0x2000 的偏移量即可。

uDSC 运算指令定义

软件通过 uDSC 的 IR 寄存器指定需要实现的操作。uDSC 的所有运算操作都在 DX/DY/DA 之间进行。用户可以使用 16 位 LD/ST 通道在 DX/DY/DA 以及 SRAM 直接快速的交换数据。

分类	IR[7:0]								功能描述
ADD/SUB	0	0	S[1]	0	0	1	0	1	DA = DX + DY
	0	0	S[1]	0	0	0	0	1	DA = DX – DY
	0	0	0	1	1	1	0	1	DA = DY
	0	0	S[1]	1	1	0	0	1	DA = -DY
	0	0	S[1]	1	0	1	1	1	DA = DA + DY
	0	0	S[1]	1	0	0	1	1	DA = DA – DY
MAC/MSC	0	1	S1[2]	S0[2]	0	1	0	0	DA = DX * DY
	0	1	S1[2]	S0[2]	0	0	0	0	DA = -DX * DY
	0	1	S1[2]	S0[2]	1	1	0	0	DA = (DX * DY) >> 1
	0	1	S1[2]	S0[2]	1	0	0	0	DA = (-DX * DY) >> 1
	0	1	S1[2]	S0[2]	0	1	1	S	DA = DA + DX * DY
	0	1	S1[2]	S0[2]	1	1	1	S	DA = (DA + DX * DY) >> 1
	0	1	S1[2]	S0[2]	0	0	1	S	DA = DA – DX * DY
	0	1	S1[2]	S0[2]	1	0	1	S	DA = (DA – DX * DY) >> 1
MISC	1	0	0	0	0	0	0	0	DA = 0

	1	0	0	0	0	1	0	S	DA = NEG(DA)
	1	0	0	0	1	0	0	S	DA = DX^2
	1	0	0	0	1	0	1	S	DA = DY^2
	1	0	1	0	0	0	0	S	DA = ABS(DA)
	1	0	1	1	0	0	0	0	DA = DA/DY
	1	0	1	1	0	0	0	1	DA = DA/DY, DY = DA%DY
SHIFT	1	1	0	0	N3	N2	N1	N0	DA = DA << N
	1	1	S	1	N3	N2	N1	N0	DA = DA >> N

说明：

1. S 表示当然运算是有符号运算还是无符号运算
2. S1 表示 DX 是否为有符号数， S2 表示 DY 是否为有符号数
3. N3...0 为四位移位位数，可以实现最多 15 位移位操作
4. – 表示此位的值不无意义，可设置为 0 或 1，建议设置为 0

寄存器定义

名称	IO 地址	功能描述
DCSR	0x20(0x00)	uDSC 控制状态寄存器
DSIR	0x21(0x01)	运算指令寄存器
DSSD	0x22(0x02)	累加器 DSA 的 16 位饱和运算结果
DSDX	0x10(0x30)	操作数 DSDX, 16 位读写访问
DSDY	0x11(0x31)	操作数 DSDY, 16 位读写访问
DSAL	0x38(0x58)	32 位累加器 DSA[15:0], 16 位读写访问
DSAH	0x39(0x59)	32 位累加器 DSA[31:16], 16 位读写访问

DSCR - 控制状态寄存器

DSCR – uDSC 控制状态寄存器								
地址: 0x20 (0x00)						默认值: 0010_xxxx		
Bit	7	6	5	4	3	2	1	0
Name	DSUEN	MM	D1	D0	-	N	Z	C
R/W	R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Bit	Name	描述
7	DSUEN	uDSC 模块使能控制; 1 = 使能， 0 = 禁用
6	MM	uDSC 寄存器映射模式；详细定义请参考 16 位工作模式的介绍。 0 = 快速访问模式， 1 = IO 映射模式
5	D1	除法运算完成标志， 1 = 运算完成
4	D0	除法运算除 0 标志位
3	-	Unimplemented
2	N	运算结果为负数标志位
1	Z	运算结果为零标志位
0	C	32 加法器进位/借位标志

DSIR – 运算指令寄存器

DSIR – uDSC 运算指令寄存器								
地址: 0x21 (0x01)						默认值: 0000_0000		
Bit	7	6	5	4	3	2	1	0
Name	DSIR[7:0]							
R/W	R/W							
Bit	Name	描述						
7:0	IR	uDSC 运算指令。请参”运算指令定义“章节的描述						

DSDX - 操作数寄存器 DSDX

DSDX – uDSC 操作数寄存器 DX																
地址: 0x30 (0x10)											默认值: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSDX[15:0]															
R/W	R/W															
Bit	Name	描述														
15:0	DSDX	16 位操作数寄存器 DSDX														

DSDY - 操作数寄存器 DSDY

DSDY – uDSC 操作数寄存器 DY																
地址: 0x31 (0x11)											默认值: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSDY[15:0]															
R/W	R/W															
Bit	Name	描述														
15:0	DSDY	16 位操作数寄存器 DSDY														

DSAL - 32 位累加器 DA 的低 16 位

DSAL – uDSC 操作数寄存器 DSA 的低 16 位																
地址: 0x58 (0x38)											默认值: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSA[15:0]															
R/W	R/W															
Bit	Name	描述														
15:0	DSAL	32 位累加器 DSA 的低 16 位														

DSAH - 32 位累加器 DA 的高 16 位

DSAH – uDSC 操作数寄存器 DSA 的高 16 位																
地址: 0x59 (0x39)											默认值: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSA[31:16]															
R/W	R/W															
Bit	Name	描述														
15:0	DSAH	32 位累加器 DSA 的高 16 位														

DSSD - DA 饱和运算寄存器

DSSD – 16 位 DA 饱和运算结果																
地址: 0x22 (0x02)											默认值: 0000_0000					
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSSD[15:0]															
R/W	R/W															
Bit	Name	描述														
15:0	DSSD	32 位累加器 DSA 的 16 位饱和运算结果														

uDSC 应用实例

实例 1. 基本配置与运算

下面为一个简单子程序(AVRGCC)，实现一个 16 位的乘法运算，返回 32 位结果.:

```
unsigned long dsu_xmuluu (unsigned short dy, unsigned short dx);
```

以下为该 C 函数的汇编实现代码:

```
#include    “udsc_def.inc”          ; opcode definitions
            .global     dsu_xmuluu  ; declare for called from C/C++ code

dsu_xmuluu:
            out         DSDX, r24   ; load DX
            out         DSDY, r22   ; load DY
            ldi         r20, XMULUU ; load opcode
            out         DSIR, r20   ; do multiply
            in          r22, DSAL   ; {r23, r22} = AL
            in          r24, DSAH   ; {r25, r24} = AH
            ret
```

通用可编程端口(GPIO)

概述

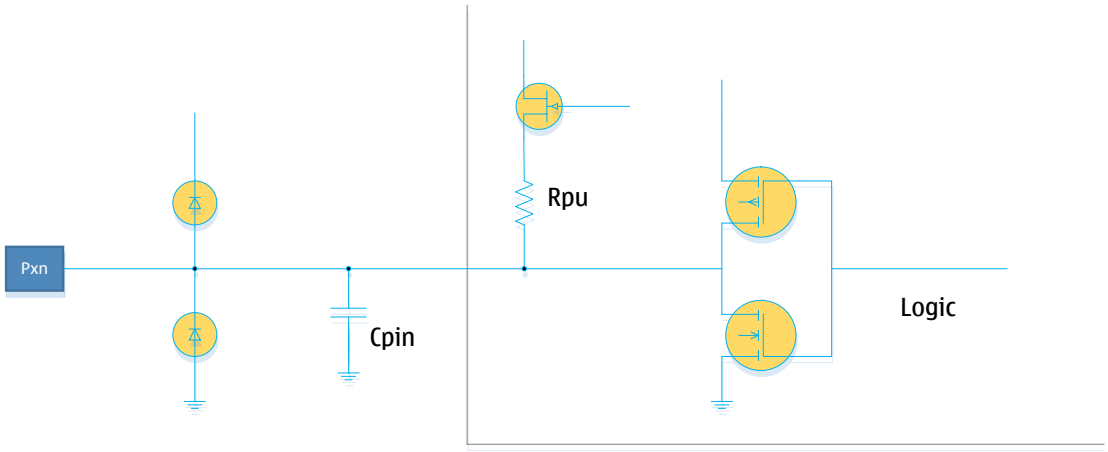
所有基于 LGT8XM 内核系列实现的 MCU 都具有 I/O 端口读-改-写功能。这意味着，某一个端口的状态可以使用 SBI 和 CBI 指令单独的改变，而不会影响到其他任何 I/O。同样，改变一个端口的方向或者控制它的上拉电阻也可以如此。

LGT8FX8P 的大部分 I/O 拥有对称的驱动特性，能够驱动和吸收较大的电流。I/O 具有两级驱动能力，用户可以控制每组 I/O 的驱动能力。I/O 的驱动能力可以直接驱动一些 LED。

LGT8FX8P 的大部分 I/O 可以驱动高达 30mA 的电流，可直接用于驱动段码 LED。

所有的 I/O 的 VCC 和 GND 直接都有独立的 ESD 保护二极管，设计至少可以承受高达 5000V 的 ESD 脉冲。

I/O 等效电路图：



本章下面所有寄存器采用统一描述方式，小写的"x"表示端口的字母序号名，小写的"n"表示端口中的位号。但当在程序中使用端口寄存器时，必须使用准确的寄存器名字。比如 PORTB3，它表示 PORTB 的第三位，这里则统一用 PORTxn 表示。I/O 相关寄存器的详细定义，请参考寄存器描述部分。

每个端口分配有三个 I/O 寄存器空间，它们为：端口数据输出寄存器(PORTx)，端口方向寄存器(DDRx)，端口数据输入寄存器(PINx)。端口数据输入寄存器为只读寄存器。数据输出寄存器与端口方向寄存器可读也可以改写。MCUCR 寄存器中的 PUD 位，用于控制所有 I/O 的上拉电阻，当 PUD 位为 1 时，将禁止所以 I/O 的上拉电阻。

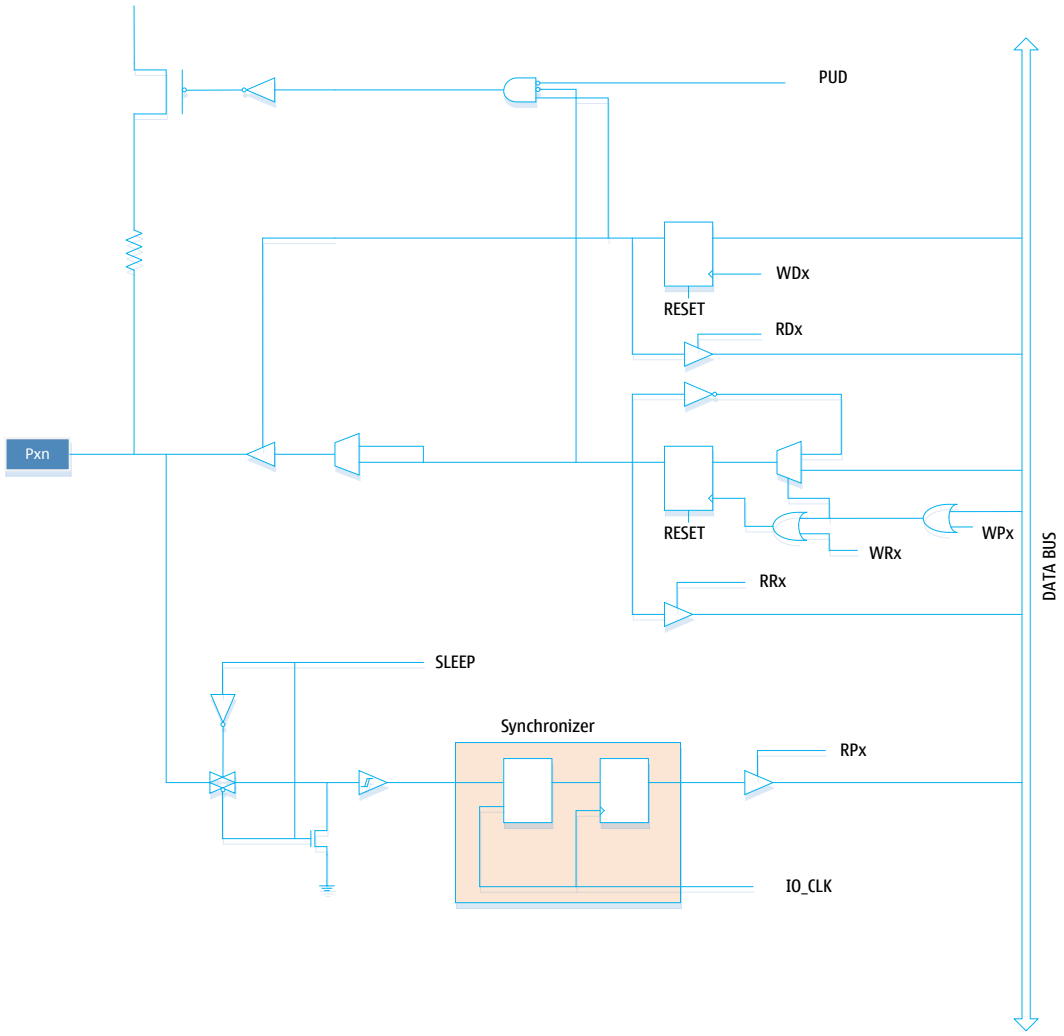
大部分 I/O 除了具有通用输入/输出功能，也会被复用为其他外设功能。具体的复用功能请参考关于端口功能复用的章节。

需要注意的是，使能某些端口的复用功能并不会影响这些端口作为数字 I/O 使用。而且某些复用功能也可能需要通过 I/O 寄存器控制端口的输入/输出方向。具体的设置将会在各个复用模块的文档的介绍。

通用输入/输出端口

作为通用 I/O 时，端口为双向驱动 I/O 端口，内部可编程上拉。

下图为通用 I/O 端口的等效电路图：



PUD: PULLUP DISABLE

SLEEP: SLEEP CONTROL

IO_CLK: I/O CLOCK

WDx: WRITE DDRx

RDx: READ DDRx

WRx: WRITE PORTx

RRx: READ PORTx REGISTER

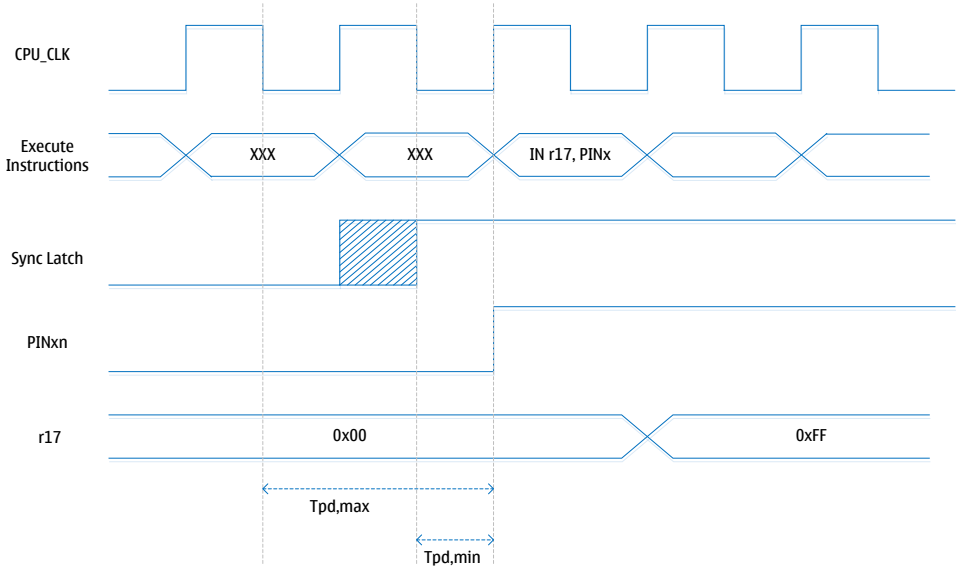
RPx: READ PORTx PIN

WPx: WRITE PINx REGISTER

端口使用配置

每个端口由三个寄存器位控制：DDxn，PORTxn 和 PINxn。其中 DDxn 用于可以通过 DDRx 寄存器访问，PORTxn 可以通过 PORTx 寄存器访问，PINxn 可以通过 PINx 寄存器访问。

DDRxn 寄存器位用于设置端口的输入/输出方向。如果 DDxn 设置为 1，Pxn 端口就被配置为一个输出端口。如果 DDxn 设置为 0，Pxn 就被配置为一个输入端口。



如果 PORTxn 位被写 1，同时这个端口被配置为输入端口，这个端口的上拉电阻有效。如果想要禁止端口的上拉电阻，PORTxn 必须写为 0 或者将这个端口配置为输出端口。

端口的复位初始化状态为输入状态，上拉电阻无效。

PORTxn 设置为 1，同时这个端口被配置为输出端口，外部端口将会被驱动为高电平。如果 PORTxn 设置为 0，端口将会被驱动为低。

输入/输出切换

当 I/O 状态在三态([DDxn, PORTxn]) = 0b00)和输出高电平([DDxn, PORTxn] = 0b11)之间切换时，将会出现一个端口上拉或者输出为低的中间状态。通常，上拉电阻是可以被接受的，因为在一个高阻环境下，驱动为高和上拉之间的区别并不重要。如果不是这种情况，可以通过 MCUCR 寄存器中的 PUD 位关闭所以端口的上拉功能。

同样，在上拉使能的输入与输出低电平之间切换时，也会出现同样的问题。用户必须使用三态([DDxn, PORTxn] = 0b00)或者输出高([DDxn, PORTxn] = 0b11)作为中间状态。

端口驱动配置表：

DDxn	PORTxn	PUD	端口状态	上拉	功能说明
0	0	X	输入	禁止	三态(High-Z)
0	1	0	输入	使能	输入+内部上拉模式
0	1	1	输入	禁止	三态(High-Z)
1	0	X	输出	禁止	输出低(扇入)
1	1	X	输出	禁止	输出高(扇出)

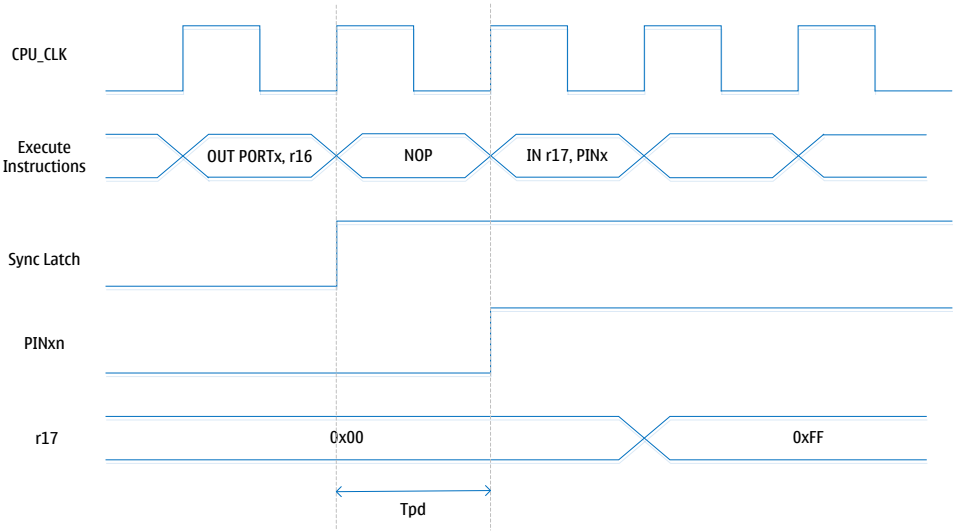
读端口值

无论端口方向位 DDxn 如何设置，都可以通过 PINxn 寄存器位读取到端口的当前状态。为避免直接读取端口产生的亚稳态，PINxn 寄存器位是端口经过一个同步器的结果。同步器为一个锁存器和一个寄存器共同组成，因此 PINxn 的值与当前端口之间有一个很小的延迟。这个延迟是因为同步器存在的结果，延迟时间最多为 1 个半系统周期。

我们假设系统周期从系统时钟的第一个下降沿开始，锁存器在时钟为低的时候锁存数据，时钟为高时数据直通过锁存器，如上图中阴影部分所示。在时钟为低电平时，端口数据被锁

存，并且在下一个时钟的上升沿被寄存器到 PINxn 寄存器。上图中的 Tpd,max 以及 Tpd,min 为端口数据的最大和最小延迟，分为为 1.5 周期和 0.5 周期。

如果要读取到软件设置的端口值，需要在 I/O 的写和读字节支持插入一个空操作指令(NOP)。时序如下图所示：



下面的代码说明如何设置端口 B 的引脚 0/1 为高，2/3 为低，定义引脚 4~7 为输入并且使能了引脚 6、7 的上拉电阻。然后引脚的值回读到通用工作寄存器中，按照之前的描述，在引脚的输出和输入直接插入了一个 NOP 指令。

汇编代码

```
; Define Pull-ups and set outputs high
; Define directions for port pins
LDI r16, (1<<PB7)|(1<<PB6)|(1<<PB1)|1<<PB0)
LDI r17, (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
OUT PORTB, r16
OUT DDRB, r17
; Insert nop for synchronization
NOP
; Read port pins
IN r16, PINB
```

C 语言代码

```
unsigned char I;
/* Define pull-ups and set outputs high */
/* Define directions for port pins */
PORTB = (1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
/* Insert nop for synchronization */
__no_operation();
/* Read port pins */
I = PINB;
```

输入使能与休眠控制

从 I/O 的等效电路图中我们可以看到，数字输入可以在 SLEEP 信号的控制下被钳位到地电平。SLEEP 信号由 MCU 的休眠控制器以及各种休眠模式控制。这样可以保证在进入休眠后，系统不会因为端口输入浮空而造成漏电。

端口的 SLEEP 控制作用会被外部中断功能取代。如果外部中断请求无效，SLEEP 控制仍然可以起作用。SLEEP 控制功能也会被其他一些第二功能取代，具体请参考下面关于端口第二功能的介绍。

快速翻转端口状态

端口状态设置为输出的 IO，可以通过 PORTn 寄存器改变端口状态。如果需要翻转当前端口的输出状态，通常需要首先读取当前端口状态 PINx，然后取反回写到 PORTn 寄存器完成翻转。LGT8FX8P 提供另外一种更加高效的方式翻转端口状态，通过直接向 PINx 寄存器写 1 即可实现将指定的端口状态翻转。比如我们写 PINB[3]为 1，即可实现将 PB3 的端口状态翻转。对于需要产生输出时钟的应用中，这种方式非常的实用。

数字/模拟复用端口

LGT8FX8P 部分端口为数模功能混合复用端口。除内部 DAC 的输出 PD4 外，其他混合端口的均作为模拟输入用。当端口作为模拟功能使用时，软件需要将该端口设置为输入模式，并根据需要关闭内部上拉，以免对模拟收入产生影响。DIDR0~2 寄存器用于关闭混合功能端口的数字输入通道，以避免模拟输入对数字电路造成多余功耗损失。DIDRx 不会关闭端口的数字输出功能。

大电流推挽驱动端口

LGT8FX8P 支持对多 6 路大电流推挽驱动端口，支持最大 80mA 的推挽驱动。考虑到芯片 VCC 最大过电流能力限制，不建议同时开启 6 路大电流驱动。特别是对于只有一组电源端口的 QFP32 封装，建议不要同时开启并驱动 4 路以上的大电流负载。

普通端口的驱动为 12mA，软件需要通过 HDR 寄存器开启端口的大电流驱动功能。

具备大电流驱动能力的端口如下：

HDR 端口	QFP48	QFP32	HDR	功能说明
PD5	PD5	PD5	HDR[0]	N/A
PD6	PD6	PD6	HDR[1]	N/A
PF1	PF1	PD1 PF1	HDR[2]	QFP32 封装的 PD1 内部相当于 QFP48 的 PD1 与 PF1 并联
PF2	PF2	PD2 PF2	HDR[3]	QFP32 封装的 PD2 内部相当于 QFP48 的 PD2 与 PF2 并联
PF4	PF4	PE4 PF4	HDR[4]	QFP32 封装的 PE4 内部相当于 QFP48 的 PF4 与 PE4 并联
PF5	PF5	PE5 PF5	HDR[5]	QFP32 封装的 PE5 内部相当于 QFP48 的 PF5 与 PE5 并联

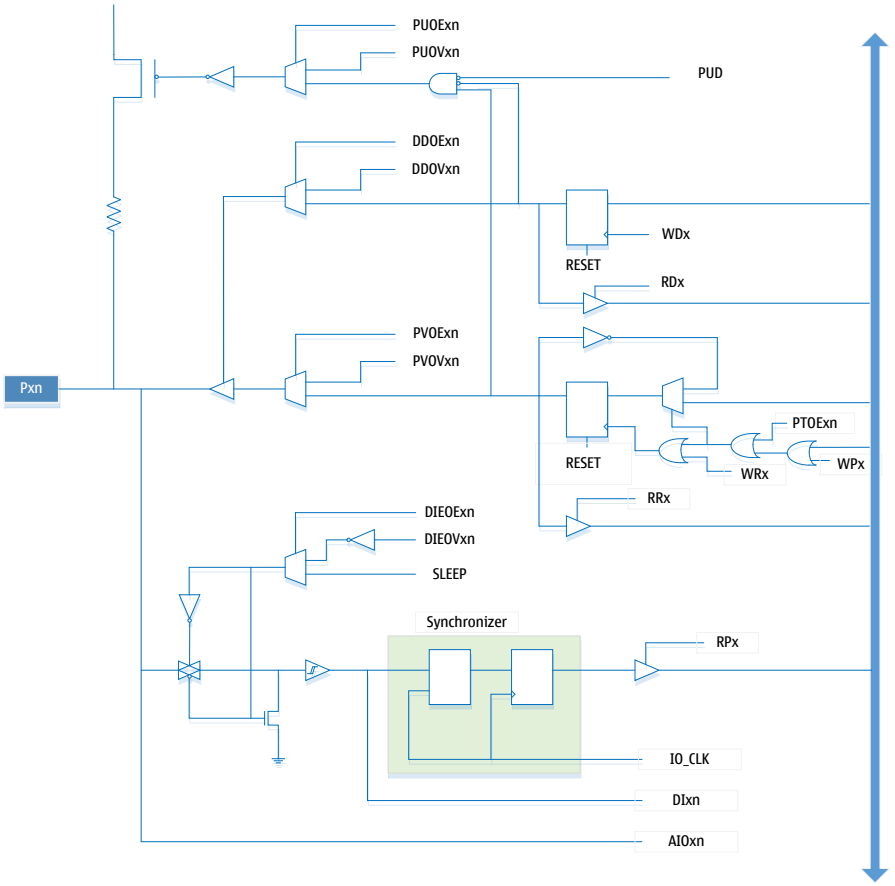
空闲端口的处理

如果一些端口没有被使用，建议将他们驱动到一个固定的电平。在任何情况下，浮空的引脚都会带来更多的功耗，并且会导致系统在强干扰下变的不稳定。

给端口一个固定电平最简单的方法就是打开端口的上拉电阻。需要注意的是，上拉电阻在上电复位过程中是禁止的。上拉电阻的方式也会带来多余的漏电。因此建议使用外部的上拉或者下拉电阻连接。直接将端口与电源或地连接是不建议的，因为如果这些引脚被配置为输出，会有可能导致非常大的电流由端口经过，对芯片造成破坏性的影响。

端口复用功能

大部分端口都有复用功能，下面的等效电路说明了端口复用功能对端口的控制。这些复用功能并不一定存在与所以的端口引脚。



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE	PUD:	PULLUP DISABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE	WDx:	WRITE DDRx
DDOExn:	Pxn DATA DIRECTION OVERRRIDE ENABLE	RDx:	READ DDRx
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RRx:	READ PORTx REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WRx:	WRITE PORTx
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	RPx:	READ PORTx PIN
DIEOExn:	Pxn INPUT-ENABLE OVERRIDE ENABLE	WPx:	WRITE PINx
DIEOVxn:	Pxn INPUT-ENABLE OVERRIDE VALUE	IO_CLK:	I/O CLOCK
SLEEP:	SLEEP CONTROL	DIxn:	INPUT PIN n ON PORTx
PTOExn:	Pxn PORT TOGGLE OVERRIDE ENABLE	AIOxn:	ANALOG I/O PIN n ON PORTx

复用功能控制信号一般描述：

信号	全称	功能描述
PUOE	上拉复用使能	此位为 1，上拉使能由 PVOV 控制；如果此位为 0，上拉使能受 DDxn, PORTxn 以及 PUD 共同控制
PUOV	上拉复用值	如果 PUOE 为 1，此位为 1 将使能引脚的上拉电阻，否则将禁止引脚上拉电阻
DDOE	端口方向复用使能	次位为 1，引脚输出使能由 DDOE 控制，否则由 DDxn 控制
DDOV	端口方向复用值	如果 DDOE 为 1，次位为 1，将使能引脚的输出功能，否则关闭引脚的输出
PVOE	端口数据复用使能	如果次位为 1，并且引脚输出使能，引脚的输出值将由 PVOV 控制，否则是由 PORTxn 控制
PVOV	端口数据复用值	参考 PVOE 功能描述
PTOE	端口翻转复用使能	次位为 1，PORTxn 位将翻转
DIEOE	数字输入使能复用使能	如果次位为 1，端口数字输入使能将由 DIEOV 控制；否则将有 MCU 的运行状态控制
DIEOV	数字输入使能复用值	如果 DIEOE 为 1，端口的数字输入功能将由次位控制，与 MCU 运行状态无关
DI	数字输入	这个是输入给替代功能模块的数字输入信号。从 I/O 等下电路图中可以看到，这个值在施密特触发器之后，但在 I/O 输入同步器之前。这个信号连接到外设模块中，外设模块将会根据需要进行同步处理
AIO	模拟输入	模拟输入/输出信号，这个信号直接与 I/O 的 PAD 相连，可作为模拟的双向信号使用。这个信号直接与内部的 ADC、比较器等模拟模块的端口相连接

下面一小节将会简短的描述每个引脚的复用功能和相关的控制信号。

端口 B 复用功能

引脚	复用功能描述
PB7	XTALI/TOSC2 (外部主晶振引脚 XI) PCINT7 (引脚电平变化中断 7)
PB6	XTALO/TOSC1 (外部主晶振引脚 XO) PCINT6 (引脚电平变化中断 6)
PB5	SCK (SPI 总线主时钟输入) PCINT5 (引脚电平变化中断 5)
PB4	MISO (SPI 总线主输入/从输出) PCINT4 (引脚电平变化中断 4)

PB3	MOSI (SPI 总线主输出/从输入) OC2A (定时/计数器 2 比较匹配输出 A) PCINT3 (引脚电平变化中断 3)
PB2	SSN (SPI 总线从设备选择输入) OC1B (定时/计数器 1 比较匹配输出 B) PCINT2 (引脚电平变化中断 2)
PB1	OC1A (定时/计数器 1 比较匹配输出 A) PCINT1 (引脚电平变化中断 1)
PB0	ICP1 (定时/计数器 1 俘获输入) CLKO (系统时钟输出) PCINT0 (引脚电平变化中断 0)

XTALI/TOSC2/PCINT7 – 端口 B 引脚 7

XTALI: 外部晶振引脚 XI。当用作晶振的时钟信号时，这个引脚将不能作为 I/O 使用。
TOSC2: 定时器外部晶振引脚 2。当内部 RC 被配置为芯片的主工作时钟，并且使能了异步定时器功能(ASSR 寄存器配置)，此引脚将作为定时器的外部晶振引脚。当 ASSR 寄存器的 AS2 被设置为 1，EXCLK 为设置为 0，便使能了定时/计数器 2 使用外部晶振的异步时钟功能，PB7 将与内部 I/O 端口断开，成为内部振荡放大器的反向输出引脚。这种模式下，外部晶振与引脚相连接。
PCINT7: 引脚电平变化中断 7。PB7 为外部中断源。
如果 PB7 被用于晶振引脚，DDB7,PORTB7 和 PINB7 的值将没有任何意义。

XTALO/TOSC1/PCINT6- 端口 B 引脚 6

XTALO: 外部晶振引脚 XO。
TOSC1: 定时器外部晶振引脚 1。当内部 RC 被配置为芯片的主工作时钟，并且使能了异步定时器功能(ASSR 寄存器配置)，此引脚将作为定时器的外部晶振引脚。当 ASSR 寄存器的 AS2 被设置为 1，EXCLK 为设置为 0，便使能了定时/计数器 2 使用外部晶振的异步时钟功能，PB6 将与内部 I/O 端口端口，成为内部振荡放大器的输入引脚。这种模式下，外部晶振与引脚相连接。
PCINT6: 引脚电平变化中断 6。PB6 为外部中断源。
如果 PB6 被用于晶振引脚，DDB6,PORTB6 和 PINB6 的值将没有任何意义。

SCK/PCINT5- 端口 B 引脚 5

SCK: SPI 控制器主设备时钟输出，从设备时钟输入。当 SPI 控制器被配置为一个从设备，这个引脚将被配置为一个输入引脚，不受 DDB5 的控制。当 SPI 控制器被配置为主设备，这个引脚的方向由 DDB5 控制。当这个引脚被 SPI 强制为输入后，仍然可以通过 PORTB5 位控制上拉电阻。
PCINT5: 引脚电平变化中断。PB5 为外部中断源。

MISO/PCINT4- 端口 B 引脚 4

MISO: SPI 控制主设备数据输入，从设备数据输出。当 SPI 被配置为主设备，这个引脚将会被强制为输入，并不受 DDB4 的控制。当 SPI 作为一个从设备时，这个引脚的数据方

向由 DDB4 控制。当这个引脚被 SPI 控制器强制为输入后，它的上拉电阻仍然可以通过 PROTB4 控制。

PCINT4: 引脚电平变化中断。PB4 为外部中断源。

MOSI/OC2A/PCINT3- 端口 B 引脚 3

MOSI: SPI 控制器主设备数据输出，从设备数据输入。当 SPI 被配置为从设备，这个引脚将会被强制为输入，并不受 DDB3 的控制。当 SPI 控制器被配置为主设备，这个引脚的方法由 DDB3 控制。当这个引脚被 SPI 控制强制为输入，仍然可以通过 PORTB3 控制它的上拉电阻。

OC2A: 定时/计数器 2 的 A 组比较匹配输出。PB3 可以作为定时/计数器 2 比较匹配的外部输出。此时必须通过 DDB3 将引脚设置为输出。同时，OC2A 也是定时器 2 的 PWM 模式输出引脚。

PCINT3: 引脚电平变化中断。PB3 为外部中断源。

SSN/OC1B/PCINT2- 端口 B 引脚 2

SSN: SPI 从设备片选输入。当 SPI 控制器配置为从设备，这个引脚将会被强制为输入，并不受 DDB2 的控制。作为一个从设备，SPI 控制器在 SSN 被驱动为低是有效。当 SPI 控制器配置为主设备，这个引脚的方向由 DDB2 控制。当这个引脚被 SPI 控制器强制为输入后，仍然可以通过 PORTB2 控制上拉电阻。

OC1B: 定时/计数器 1 的 B 组比较匹配输出。PB2 可以作为定时/计数器 1 比较匹配的外部输出。此时必须通过 DDB2 将引脚设置为输出。同时，OC1B 也是定时器 1 的 PWM 模式输出引脚。

PCINT2: 引脚电平变化中断。PB2 为外部中断源。

OC1A/PCINT1- 端口 B 引脚 1

OC1A: 定时/计数器 1 的 A 组比较匹配输出。PB1 可以作为定时/计数器 1 比较匹配的外部输出。此时必须通过 DDB1 将引脚设置为输出。同时，OC1A 也是定时器 1 的 PWM 模式输出引脚。

PCINT1: 引脚电平变化中断。PB1 为外部中断源。

ICP1/CLKO/PCINT0- 端口 B 引脚 0

ICP1: 定时/计数器 1 的俘获输入引脚

CLKO: 系统工作时钟输出，当 CLKPR 寄存器中的 CLKOE 位为 1，这个引脚将会被强制为输出，不受 DDB0 的控制。输出频率为当前系统的工作时钟频率。

PCINT0: 引脚电平变化中断。PB0 为外部中断源。

端口 C 复用功能

引脚	复用功能描述
PC7	ADC8(ADC 输入通道 8) APN2(DAP 反向输入 2) PCINT15(引脚电平变化输入 15)
PC6	RESETN (外部复位输入) PCINT14 (引脚电平变化输入 14)
PC5	ADC5 (ADC 输入通道 5) SCL (TWI 时钟线) PCINT13 (引脚电平变化输入 13)
PC4	ADC4 (ADC 输入通道 4) SDA (TWI 数据线) PCINT12 (引脚电平变化输入 12)
PC3	ADC3 (ADC 输入通道 3) PCINT11 (引脚电平变化输入 11)
PC2	ADC2 (ADC 输入通道 2) PCINT10 (引脚电平变化输入 10)
PC1	ADC1 (ADC 输入通道 1) PCINT9 (引脚电平变化输入 9)
PC0	ADC0 (ADC 输入通道 0) PCINT8 (引脚电平变化输入 8)

ADC8/APN2/PCINT15- 端口 C 引脚 6

ADC8: ADC 外部输入通道 8

APN2: 差分放大器的反向输入端口 2

PCINT15: 引脚电平变化中断。关闭这个引脚的外部复位输入功能后， PC7 可以做为外部中断源。

RESETN/PCINT14- 端口 C 引脚 6

RESETN: 外部复位输入引脚。上电复位后，这个引脚默认为外部复位功能。可以通过 IOCR 寄存器关闭外部复位功能。关闭外部复位功能后，这个引脚可作为通用 I/O 使用。但需要注意的是，在上电和其他复位过程中，这个引脚默认为复位输入，所以如果用户需要用到这个引脚的通用 I/O 功能，外部电路不能影响到芯片的上电和复位过程，建议将这个引脚配置为输出功能的 I/O，并在外部加一个适当的上拉电阻。

PCINT14: 引脚电平变化中断。关闭这个引脚的外部复位输入功能后， PC6 可以做为外部中断源。

SCL/ADC5/PCINT13- 端口 C 引脚 5

SCL: TWI 接口时钟信号。TWCR 寄存器中的 TWEN 位置 1 后，使能 TWI 接口，PC5 将被 TWI 控制，成为 TWI 接口的时钟信号。

ADC5: ADC 输入通道 5。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部

分对模拟电路的影响。具体请参考 ADC 相关章节。
PCINT13: 引脚电平变化中断 13

SDA/ADC4/PCINT12- 端口 C 引脚 4

SDA: TWI 接口数据信号。TWCR 寄存器中的 TWEN 位置 1 后，使能 TWI 接口，PC4 将被 TWI 控制，成为 TWI 接口的数据信号。
ADC4: ADC 输入通道 4。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节。
PCINT12: 引脚电平变化中断 12

ADC3/APN1/PCINT11- 端口 C 引脚 3

ADC3: ADC 输入通道 3。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节。
APN1: 差分放大器反向输入 1
PCINT11: 引脚电平变化中断 11

ADC2/APN0/PCINT10- 端口 C 引脚 2

ADC2: ADC 输入通道 2。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节。
APN0: 差分放大器反向输入 0
PCINT10: 引脚电平变化中断 10

ADC1/APP1/PCINT9- 端口 C 引脚 1

ADC1: ADC 输入通道 1。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节。
APP1: 差分放大器正向输入 1
PCINT9: 引脚电平变化中断 9

ADC0/APP0/PCINT8- 端口 C 引脚 0

ADC0: ADC 输入通道 0。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节。
APP0：差分放大器正向输入 0
PCINT8: 引脚电平变化中断 8

端口 D 复用功能

引脚	复用功能描述
PD7	ACXN (模拟比较器 0/1 公用负端输入) PCINT23 (引脚电平变化中断 23)
PD6	ACOP (QFP32: 模拟比较器 0 正端输入) OC0A (定时/计数器 0 比较匹配输出 A) OC3A (QFP32: 定时/计数器 3 比较匹配输出 A) PCINT22 (引脚电平变化中断 22)
PD5	T1 (定时/计数器 1 外部计数时钟输入) OC0B (定时/计数器 0 比较匹配输出 B) PCINT21 (引脚电平变化中断 21)
PD4	XCK (USART 外部时钟输入/输出) DAO(内部 8bit DAC 模拟输出) T0 (定时/计数器 0 外部计数时钟输入) PCINT20 (引脚电平变化中断 20)
PD3	INT1 (外部中断输入 1) OC2B (定时/计数器 2 比较匹配输出 B) PCINT19 (引脚电平变化中断 19)
PD2	INT0 (外部中断输入 0) ACO0 (比较器 0 输出) OC3B (QFP32: 定时/计数器 3 比较匹配输出 B) PCINT18 (引脚电平变化中断 18)
PD1	TXD (USART 数据输出) OC3A (QFP32: 定时/计数器 3 比较匹配输出 A) PCINT17 (引脚电平变化中断 17)
PD0	RXD (USART 数据输入) PCINT16 (引脚电平变化中断 16)

ACXN/OC2B/PCINT23- 端口 D 引脚 7

ACXN: 模拟比较器 0/1 公用负端输入

OC2B: 定时/计数器 2 的 B 组比较匹配输出。PD7 可以作为定时/计数器 2 比较匹配的外部输出。此时必须通过 DDD7 将引脚设置为输出。同时，OC2B 也是定时器 2 的 PWM 模式输出引脚;

PCINT23: 引脚电平变化中断 23

ACOP/OC0A/PCINT22- 端口 D 引脚 6

ACOP: 模拟比较器 0 正端输入。

OC0A: 定时/计数器 0 的 A 组比较匹配输出。PD6 可以作为定时/计数器 0 比较匹配的外部输出。此时必须通过 DDD6 将引脚设置为输出。同时，OC0A 也是定时器 0 的 PWM 模式输出引脚

PCINT22: 引脚电平变化中断 22

T1/OC0B/PCINT21- 端口 D 引脚 5

T1: 定时/计数器 1 的外部计数时钟输入
OC0B: 定时/计数器 0 的 B 组比较匹配输出。PD5 可以作为定时/计数器 0 比较匹配的外部输出。此时必须通过 DDD5 将引脚设置为输出。同时，OC0B 也是定时器 0 的 PWM 模式输出引脚
PCINT21: 引脚电平变化中断 21

XCK/T0/DAO/PCINT20- 端口 D 引脚 4

XCK: 同步模式 USART 的外部时钟信号
T0: 定时/计数器 0 的外部计数时钟输入
DAO: 内部 8 位 DAC 模拟输出
PCINT20: 引脚电平变化中断 20

INT1/OC2B/PCINT19- 端口 D 引脚 3

INT1: 外部中断输入 1
OC2B: 定时/计数器 2 的 B 组比较匹配输出。PD3 可以作为定时/计数器 2 比较匹配的外部输出。此时必须通过 DDD3 将引脚设置为输出。同时，OC2B 也是定时器 2 的 PWM 模式输出引脚
PCINT19: 引脚电平变化中断 19

INT0/OC3B/AC00/PCINT18- 端口 D 引脚 2

INT0: 外部中断输入 0
OC3B: 定时计数器 3 比较匹配输出 B。仅在 QFP32 封装时，PD2 与 QFP48/PF2 合并成一个 IO，因此 PF2 上的 OC3B 功能也将从 PD2 上输出
AC00: 模拟比较器 0 比较结果直接输出。由 AC0FR 寄存器控制
PCINT18: 引脚电平变化中断 18

TXD/OC3A/PCINT17- 端口 D 引脚 1

TXD: 传输数据(USART 数据输出)。USART 发送器使能后，PD1 将被强制为输出，不受 DDD1 的控制
OC3A: 定时计数器 3 比较匹配输出 A。仅在 QFP32 封装时，PD1 与 QFP48/PF1 合并成一个 IO，因此 PF1 上的 OC3A 功能也将从 PD1 上输出
PCINT17: 引脚电平变化中断 17

RXD/PCINT16- 端口 D 引脚 0

RXD: 传输数据(USART 数据输入)。USART 接收器使能后，PD0 将被强制为输入，不受 DDD0 的控制。当引脚被 USART 强制为输入后，上拉电阻仍然可以通过 PORTD0 位控制
PCINT16: 引脚电平变化中断 16

端口 E 复用功能

引脚	复用功能描述
PE7	ADC11 (ADC 输入通道 11) PCINT31 (引脚电平变化中断 31)
PE6	AVREF (QFP32: ADC 外部参考电压) ADC10 (ADC 输入通道 10) PCINT30 (引脚电平变化中断 30)
PE5	CLKO (系统时钟输出) AC1O (模拟比较器 1 输出) PCINT29 (引脚电平变化中断 29)
PE4	OC0A (定时/计数器 0 比较配置输出 A) PCINT28 (引脚电平变化中断 28)
PE3	ADC7 (ADC 输入通道 7) AC1N (模拟比较器 1 负端输入) PCINT27 (引脚电平变化中断 27)
PE2	SWD (SWD 调试器数据线) PCINT26 (引脚电平变化中断 26)
PE1	ADC6 (ADC 输入通道 6) ACXP (模拟比机器 0/1 公用正端输入) PCINT25 (引脚电平变化中断 25)
PE0	SWC (SWD 调试器时钟输入) APN4 (差分放大器反向输入 4) PCINT24 (引脚电平变化中断 24)

ADC11/PCINT31- 端口 E 引脚 7

ADC11: ADC 外部输入通道 11
PCINT31: 引脚电平变化中断 30

AVREF/ADC10/PCINT30- 端口 E 引脚 6

AVREF: ADC 外部参考电源输入，用作模拟功能时，需要将对应的数字 I/O 设置为输入，并关闭上拉电阻，以避免数字电路对模拟电路产生干扰
ADC10: ADC 模拟输入通道 10
PCINT30: 引脚电平变化中断 30

CLKO/AC1O/PCINT29- 端口 E 引脚 5

CLKO: 此功能与 PB0 的 CLKO 功能相同。可作为 PB0/CLKO 的备用引脚
AC1O: 模拟比较器 1 输出
PCINT29: 引脚电平变化中断 29

OC0A/PCINT28- 端口 E 引脚 4

OC0A: 定时/计数器 0 的 A 组比较匹配输出。PE4 可以作为定时/计数器 0 比较匹配的外部输出。此时必须通过 DDE4 将引脚设置为输出。同时，OC0A 也是定时器 0 的 PWM 模式输出引脚。
PCINT28: 引脚电平变化中断 28

ADC7/ AC1N/PCINT27- 端口 E 引脚 3

ADC7: ADC 输入通道 7。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节
AC1N: 模拟比较器 1 负端输入
PCINT27: 引脚电平变化中断 27

SWD/PCINT26- 端口 E 引脚 2

SWD: SWD 调试器数据线。PE2 默认为 SWD 功能。用户可以通过将 MCUSR 寄存器 SWDD 位置 1 关闭 SWD 调试器功能。SWD 被关闭后，调试功能将不能使用。
PCINT26: 引脚电平变化中断 26

ADC6/ACXP/PCINT25- 端口 E 引脚 1

ADC6: ADC 输入通道 6。DIDR 寄存器用于关闭数模复用 I/O 的数字功能，以避免数字部分对模拟电路的影响。具体请参考 ADC 相关章节
ACXP: 模拟比较器 0/1 公用正端输入
PCINT25: 引脚电平变化中断 25

SWC/APN4/PCINT24- 端口 E 引脚 0

SWC: SWD 调试器时钟线。PE0 默认为 SWC 功能。用户可以通过将 MCUSR 寄存器 SWDD 位置 1 关闭 SWD 调试器功能。SWD 被关闭后，调试功能将不能使用
APN4: 差分放大器反向输入 4
PCINT24: 引脚电平变化中断 24

端口 F 复用功能

引脚	复用功能描述
PF7	OC2B (定时/计数器 2 比较匹配输出 B) PCINT39 (引脚电平变化中断 39)
PF6	T3 (定时/计数器 3 外部时钟输入) OC2A (定时/计数器 2 比较匹配输出 A) PCINT38 (引脚电平变化中断 38)
PF5	OC1A (定时/计数器 1 比较匹配输出 A) PCINT37 (引脚电平变化中断 37)
PF4	OC1B(定时/计数器 1 比较配置输出 B) ICP3 (定时/计数器 3 外部俘获输入) PCINT36 (引脚电平变化中断 36)
PF3	OC0B (定时/计数器 0 比较配置输出 B) PCINT35 (引脚电平变化中断 35)
PF2	OC3B (定时/计数器 3 比较匹配输出 B) PCINT34 (引脚电平变化中断 34)
PF1	OC3A (定时/计数器 3 比较匹配输出 A) PCINT33 (引脚电平变化中断 33)
PF0	ADC9 (ADC 外部输入通道 9) APN3 (差分放大器反向输入 3) PCINT32 (引脚电平变化中断 32)

OC2B/PCINT39 - 端口 F 引脚 7

OC2B: 定时/计数器 2 比较匹配输出 B。输出选择受 PMX1 寄存器控制
PCINT39: 引脚电平变化中断 39

OC2A/T3/PCINT38 - 端口 F 引脚 6

OC2A: 定时/计数器 2 比较匹配输出 A。输出选择受 PMX1 寄存器控制
T3: 定时/计数器 3 外部时钟输入
PCINT38: 引脚电平变化中断 38

OC1A/PCINT37 - 端口 F 引脚 5

OC1A: 定时/计数器 1 比较匹配输出 A。输出选择受 PMX0 寄存器控制
PCINT37: 引脚电平变化中断 37

ICP3/OC1B/PCINT36 - 端口 F 引脚 4

OC1B: 定时/计数器 1 的 B 组比较匹配输出。输出选择受 PMX0 寄存器控制
ICP3: 定时/计数器 3 外部俘获输入
PCINT36: 引脚电平变化中断 36

OC3C/OC0B/PCINT35- 端口 F 引脚 3

OC0B: 定时/计数器 0 的 B 组比较匹配输出。输出选择受 PMX0 寄存器控制
OC3C: 定时/计数器 3 的 C 组比较匹配输出
PCINT35: 引脚电平变化中断 35

OC3B/PCINT34- 端口 F 引脚 2

OC3B: 定时/计数器 3 的 B 组比较匹配输出
PCINT34: 引脚电平变化中断 34

OC3A/PCINT33- 端口 F 引脚 1

OC3A: 定时/计数器 3 的 B 组比较匹配输出。输出选择受 PMX1 寄存器控制
PCINT33: 引脚电平变化中断 33

ADC9/APN3/PCINT32- 端口 F 引脚 0

ADC9: ADC 外部模式输入通道 9
APN3: 差分放大器反向输入 3
PCINT32: 引脚电平变化中断 32

寄存器定义

端口 B 输出数据寄存器- PORTB

PORTB – 端口 B 输出数据寄存器								
PORTB: 0x05(0x25)				默认值: 0x00				
Bits	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PORTB	B 组端口输出寄存器						

端口 B 方向寄存器- DDRB

DDRB – 端口 B 方向寄存器								
DDRB: 0x04(0x24)				默认值: 0x00				
DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	DDB	端口 B 组方向控制位; 1 = 输出, 0 = 输入						

端口 B 输入数据寄存器- PINB

PINB – 端口 B 输入数据存器								
PINB: 0x03(0x23)				默认值: 0x00				
PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PINB	B 组端口状态寄存器。读 PINB 直接获得端口的当前状态; 写 PINBn 位 1 将翻转 PORTBn 的输出状态						

端口 C 输出数据寄存器- PORTC

PORTC – 端口 C 输出数据寄存器								
PORTC: 0x08(0x28)				默认值: 0x00				
PORTC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PORTC	C 组端口输出寄存器						

端口 C 方向寄存器- DDRC

DDRC – 端口 C 方向寄存器								
DDRC: 0x07(0x27)				默认值: 0x00				
DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	DDC	C 组端口方向控制位; 1 = 输出, 0 = 输入						

端口 C 输入数据寄存器- PINC

PINC – 端口 C 输入数据存器								
PINB: 0x06(0x26)				默认值: 0x00				
PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PINC	C 组端口状态寄存器; 读 PINC 得到当前端口状态 写 PINC 将翻转当前端口输出						

端口 D 输出数据寄存器- PORTD

PORTD – 端口 D 输出数据寄存器								
PORTD: 0x0B(0x2B)				默认值: 0x00				
Bits	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								

[7:0]	PORTD	D 组端口输出寄存器

端口 D 方向寄存器- DDRD

DDRD – 端口 D 方向寄存器								
DDRD: 0x0A(0x2A)				默认值: 0x00				
DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	DDD	D 组端口输出方向控制寄存器						

端口 D 输入数据寄存器- PIND

PIND – 端口 D 输入数据存器								
PIND: 0x09(0x29)				默认值: 0x00				
PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PIND	D 组端口状态寄存器 读 PIND 获得当前端口电平状态 写 PINDn 为 1，翻转 PORTDn 对应位的状态						

端口 E 输出数据寄存器- PORTE

PORTE – 端口 E 输出数据寄存器								
PORTE: 0x0E(0x2E)				默认值: 0x00				
Bits	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PORTE	E 组端口输出寄存器						

端口 E 方向寄存器- DDRE

DDRE – 端口 E 方向寄存器								
DDRE: 0x0D(0x2D)				默认值: 0x00				
DDRE	DDE7	DDE6	DDE5	DDE4	DDE3	DDE2	DDE1	DDE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	DDE	E 组端口方向控制寄存器						

端口 E 输入数据寄存器- PINE

PINE – 端口 E 输入数据存器								
PINE: 0x0C(0x2C)				默认值: 0x00				
PINE	PINE7	PINE6	PINE5	PINE4	PINE3	PINE2	PINE1	PINE0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PINE	E 组端口状态寄存器 读 PINE 获得当前端口电平状态 写 PINEn 为 1，翻转 PORTEn 位的状态						

端口 F 输出寄存器- PORTF

PINF – 端口 F 输入数据存器								
PORTF: 0x14(0x34)				默认值: 0x00				
Bits	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PORTF	F 组端口状态寄存器 输入模式的端口，对应位写 1 将开启内部上拉 输出模式的端口，对应位写 1 将驱动输出高电平						

端口 F 方向控制寄存器- DDRF

DDRF – 端口 F 方向控制寄存器								
DDRF: 0x13(0x33)				默认值: 0x00				
Bits	DDF7	DDF6	DDF5	DDF4	DDF3	DDF2	DDF1	DDF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	DDRF	F 组端口方向控制寄存器						

端口 F 状态寄存器- PINF

PINF – 端口 F 状态寄存器								
PINF: 0x12(0x32)				默认值: 0x00				
Bits	PINF7	PINF6	PINF5	PINF4	PINF3	PINF2	PINF1	PINF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
[7:0]	PINF	F 组端口状态寄存器 读 PINF 得到端口 F 的当前电平状态 PINFn 写 1，翻转 PORTFn 对应位的状态						

端口驱动控制寄存器- HDR

HDR0 – 端口驱动控制寄存器								
HDR: 0xE0				默认值: 0x00				
Bit	-	-	HDR5	HDR4	HDR3	HDR2	HDR1	HDR0
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
位定义								

[7:6]	-	保留不用
5	HDR5	PF5 输出驱动控制; 1 = 80mA 驱动, 0 = 12mA 驱动
4	HDR4	PF4 输出驱动控制; 1 = 80mA 驱动, 0 = 12mA 驱动
3	HDR3	PF2 输出驱动控制; 1 = 80mA 驱动, 0 = 12mA 驱动
2	HDR2	PF1 输出驱动控制; 1 = 80mA 驱动, 0 = 12mA 驱动
1	HDR1	PD6 输出驱动控制; 1 = 80mA 驱动, 0 = 12mA 驱动
0	HDR0	PD5 输出驱动控制; 1 = 80mA 驱动, 0 = 12mA 驱动

端口复用控制寄存器 0- PMX0

PMX0 – 端口复用控制寄存器 0								
PMX0: 0xEE				默认值: 0x00				
Bit	WCE	C1BF4	C1AF5	C0BF3	C0AC0	SSB1	TXD6	RXD5
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
位定义								
7	WCE	PMX0/1 更新使能控制；在更新 PMX0/1 寄存器之前，需要先写 WCE 位为 1，在之后的 6 个系统周期内完成对 PMX0/1 的更新。						
6	C1BF4	OC1B 辅助输出控制 1 = OC1B 输出到 PF4 0 = OC1B 输出到 PB2						
5	C1AF5	OC1A 辅助输出控制 1 = OC1A 输出到 PF5 0 = OC1A 输出到 PB1						
4	C0BF3	OC0B 辅助输出控制 1 = OC0B 输出到 PF3 0 = OC0B 输出到 PD5						
3	C0AC0	OC0A 辅助输出控制 OC0A 输出由 C0AC0 位与 TCCR0B 寄存器的 C0AS 位共同控制： { C0AC0, C0AS} = 00 = OC0A 输出到 PD6 01 = OC0A 输出到 PE4 10 = OC0A 输出到 PC0 11 = OC0A 同时输出到 PE4 和 PC0						
2	SSB1	SPSS 辅助输出控制 1 = SPSS 输出到 PB1 0 = SPSS 输出到 PB2						
1	TXD6	串口 TXD 辅助输出控制 1 = TXD 输出到 PD6, 0 = TXD 输出到 PD1						
0	RXD5	串口 RXD 辅助输入控制 1 = RXD 输入来自 PD5, 0 = RXD 输入来自 PD0						

端口复用控制寄存器 1- PMX1

PMX1 – 端口复用控制寄存器 1								
PMX1: 0xED					默认值: 0x00			
Bit	-	-	-	-	-	C3AC	C2BF7	C2AF6
R/W	-	-	-	-	-	R/W	R/W	R/W
位定义								
[7:3]	-	保留不用						
2	C3AC	OC3A 辅助输出控制 1 = OC3A 输出到 QFP48/AC0P 0 = OC3A 输出到 PF1						
1	C2BF7	OC2B 辅助输出控制 1 = OC2B 输出到 PF7 0 = OC2B 输出到 PD3						
0	C2AF6	OC2A 辅助输出控制 1 = OC2A 输出到 PF6 0 = OC2A 输出到 PB3						
使用说明								
PMX0/1 共享寄存器更新保护控制位 PMX0[7]，更新 PMX1 时，请参考 PMX0 寄存器对 PMX0[7]的控制说明。								

端口复用控制寄存器 2 – PMX2

PMX2 – 端口复用控制寄存器 2								
PMX2: 0xF0					默认值: 0x00			
Bit	WCE	STSC1	STSC0	-	-	XIEN	E6EN	C6EN
R/W	R/W	R/W	R/W	-	-	R/W	R/W	R/W
位定义								
[7]	WCE	PMX2 更新使能控制；在更新 PMX2 寄存器之前，需要先写 WCE 位为 1，在之后的 6 个系统周期内完成对 PMX2 的更新。						
[6]	STSC1	高速晶振 IO 启动电路控制 通过 PMCR 使能高速晶振后，STSC1 自动使能。当切换系统时钟到外部高速晶振后，STSC1 自动清楚。软件也可以在晶振稳定后，手动清除 STSC1，已关闭晶振启动电路，节省功耗。						
[5]	STSC0	低速晶振 IO 启动电路控制 通过 PMCR 使能低速晶振后，STSC0 自动使能。当切换系统时钟到外部低速晶振后，STSC0 自动清楚。软件也可以在晶振稳定后，手动清除 STSC0，已关闭晶振启动电路，节省功耗。						
[4:3]	-	保留不用						
[2]	XIEN	使能外部时钟输入，需要同时使能外部晶振						
[1]	E6EN	使能 PE6 的通用 IO 功能；默认 PE6 为 AVREF 功能						
[0]	C6EN	使能 PC6 的通用 IO 功能；默认 PC6 为外部复位输入						

引脚电平变化中断

- 40 个引脚电平变化中断源
- 5 个中断入口

综述

引脚电平变化中断由 PBn，PCn，PDn, PEn 和 PFn 引脚触发。只要引脚电平变化中断被使能，即使这些引脚配置为输出也能触发中断。这可以用来产生软件中断。

任何一个使能的 PBn 引脚翻转都会触发引脚电平中断 PCI0，使能的 PCn 引脚翻转将触发 PCI1，使能的 PDn 引脚翻转将触发 PCI2，使能的 PEn 引脚翻转将触发 PCI3。各个引脚变化中断的使能分别由 PCMSK0~4 寄存器来控制。所有的引脚电平变化中断都是异步检测的，可用作某些睡眠模式下的唤醒源。

寄存器定义

Pin Change Interrupt 寄存器列表

寄存器	地址	默认值	描述
PCICR	0x68	0x00	引脚改变中断控制寄存器
PCIFR	0x3B	0x00	引脚改变中断标志寄存器
PCMSK0	0x6B	0x00	引脚改变中断屏蔽寄存器 0
PCMSK1	0x6C	0x00	引脚改变中断屏蔽寄存器 1
PCMSK2	0x6D	0x00	引脚改变中断屏蔽寄存器 2
PCMSK3	0x73	0x00	引脚改变中断屏蔽寄存器 3
PCMSK4	0x74	0x00	引脚改变中断屏蔽寄存器 4

PCICR – 引脚改变中断控制寄存器

PCICR – 引脚改变中断控制寄存器								
地址: 0x68					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	PCIE4	PCIE3	PCIE2	PCIE1	PCIE0
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:5	-	保留。
4	PCIE4	引脚改变中断使能控制位 4。 当设置 PCIE4 位为“1”且全局中断使能时，引脚改变中断 4 被使能。 任何一个使能的 PFn 引脚的电平变化都会产生 PCI4 中断。PFn 引脚中断的使能可分别由 PCMSK4 寄存器来控制。 当设置 PCIE3 位为“0”时，引脚改变中断 3 被禁止。
3	PCIE3	引脚改变中断使能控制位 3。 当设置 PCIE3 位为“1”且全局中断使能时，引脚改变中断 3 被使能。

		任何一个使能的 PEn 引脚的电平变化都会产生 PCI3 中断。PEn 引脚中断的使能可分别由 PCMSK3 寄存器来控制。 当设置 PCIE3 位为“0”时，引脚改变中断 3 被禁止。
2	PCIE2	引脚改变中断使能控制位 2。 当设置 PCIE2 位为“1”且全局中断使能时，引脚改变中断 2 被使能。 任何一个使能的 PDn 引脚的电平变化都会产生 PCI2 中断。PDn 引脚中断的使能可分别由 PCMSK2 寄存器来控制。 当设置 PCIE2 位为“0”时，引脚改变中断 2 被禁止。
1	PCIE1	引脚改变中断使能控制位 1。 当设置 PCIE1 位为“1”且全局中断使能时，引脚改变中断 1 被使能。 任何一个使能的 PCn 引脚的电平变化都会产生 PCI1 中断。PCn 引脚中断的使能可分别由 PCMSK1 寄存器来控制。 当设置 PCIE1 位为“0”时，引脚改变中断 1 被禁止。
0	PCIE0	引脚改变中断使能控制位 0。 当设置 PCIE0 位为“1”且全局中断使能时，引脚改变中断 0 被使能。 任何一个使能的 PBn 引脚的电平变化都会产生 PCI0 中断。PBn 引脚中断的使能可分别由 PCMSK0 寄存器来控制。 当设置 PCIE0 位为“0”时，引脚改变中断 0 被禁止。

PCIFR – 引脚改变中断标志寄存器

PCIFR – 引脚改变中断标志寄存器								
地址: 0x3B						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	PCIF4	PCIF3	PCIF2	PCIF1	PCIF0
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:5	-	保留。						
4	PCIF4	引脚改变中断标志位 4。 任何一个使能的 PFn 引脚的电平变化都会置位 PCIF4。当 PCIE4 和全局中断均置位时，MCU 将会跳转至 PCI4 中断入口地址。PFn 引脚中断的使能可分别由 PCMSK4 寄存器来控制。 执行中断服务程序或往 PCIF4 位写“1”都会清零 PCIF4 位。						
3	PCIF3	引脚改变中断标志位 3。 任何一个使能的 PEn 引脚的电平变化都会置位 PCIF3。当 PCIE3 和全局中断均置位时，MCU 将会跳转至 PCI3 中断入口地址。PEn 引脚中断的使能可分别由 PCMSK3 寄存器来控制。 执行中断服务程序或往 PCIF3 位写“1”都会清零 PCIF3 位。						
2	PCIF2	引脚改变中断标志位 2。 任何一个使能的 PDn 引脚的电平变化都会置位 PCIF2。当 PCIE2 和全局中断均置位时，MCU 将会跳转至 PCI2 中断入口地址。PDn 引脚中断的使能可分别由 PCMSK2 寄存器来控制。 执行中断服务程序或往 PCIF2 位写“1”都会清零 PCIF2 位。						
1	PCIF1	引脚改变中断标志位 1。						

		任何一个使能的 PCn 引脚的电平变化都会置位 PCIF1。当 PCIE1 和全局中断均置位时，MCU 将会跳转至 PCI1 中断入口地址。PCn 引脚中断的使能可分别由 PCMSK1 寄存器来控制。 执行中断服务程序或往 PCIF1 位写“1”都会清零 PCIF1 位。
0	PCIF0	引脚改变中断标志位 0。 任何一个使能的 PBn 引脚的电平变化都会置位 PCIF0。当 PCIE0 和全局中断均置位时，MCU 将会跳转至 PCI0 中断入口地址。PBn 引脚中断的使能可分别由 PCMSK0 寄存器来控制。 执行中断服务程序或往 PCIF0 位写“1”都会清零 PCIF0 位。

PCMSK0 – 引脚改变中断屏蔽寄存器 0

PCMSK0 – 引脚改变屏蔽寄存器 0								
地址: 0x6B					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	PCINT7	引脚改变使能屏蔽位 7。 当设置 PCINT7 位为“1”时，PB7 引脚电平改变中断被使能。PB7 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT7 位为“0”时，PB7 引脚电平改变中断被禁止。
6	PCINT6	引脚改变使能屏蔽位 6。 当设置 PCINT6 位为“1”时，PB6 引脚电平改变中断被使能。PB6 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT6 位为“0”时，PB6 引脚电平改变中断被禁止。
5	PCINT5	引脚改变使能屏蔽位 5。 当设置 PCINT5 位为“1”时，PB5 引脚电平改变中断被使能。PB5 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT5 位为“0”时，PB5 引脚电平改变中断被禁止。
4	PCINT4	引脚改变使能屏蔽位 4。 当设置 PCINT4 位为“1”时，PB4 引脚电平改变中断被使能。PB4 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT4 位为“0”时，PB4 引脚电平改变中断被禁止。
3	PCINT3	引脚改变使能屏蔽位 3。 当设置 PCINT3 位为“1”时，PB3 引脚电平改变中断被使能。PB3 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT3 位为“0”时，PB3 引脚电平改变中断被禁止。
2	PCINT2	引脚改变使能屏蔽位 2。 当设置 PCINT2 位为“1”时，PB2 引脚电平改变中断被使能。PB2 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT2 位为“0”时，PB2 引脚电平改变中断被禁止。
1	PCINT1	引脚改变使能屏蔽位 1。 当设置 PCINT1 位为“1”时，PB1 引脚电平改变中断被使能。PB1 引脚

		上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT1 位为“0”时，PB1 引脚电平改变中断被禁止。
0	PCINT0	引脚改变使能屏蔽位 0。 当设置 PCINT0 位为“1”时，PB0 引脚电平改变中断被使能。PB0 引脚上的电平改变将置位 PCIF0，若 PCIE0 位和全局中断置位，将会产生 PCI0 中断。当设置 PCINT0 位为“0”时，PB0 引脚电平改变中断被禁止。

PCMSK1 – 引脚改变中断屏蔽寄存器 1

PCMSK1 – 引脚改变屏蔽寄存器 1								
地址: 0x6C					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	PCINT15	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	PCINT15	引脚改变使能屏蔽位 15。 当设置 PCINT15 位为“1”时，PC7 引脚电平改变中断被使能。PC7 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT15 位为“0”时，PC7 引脚电平改变中断被禁止。
6	PCINT14	引脚改变使能屏蔽位 14。 当设置 PCINT14 位为“1”时，PC6 引脚电平改变中断被使能。PC6 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT14 位为“0”时，PC6 引脚电平改变中断被禁止。
5	PCINT13	引脚改变使能屏蔽位 13。 当设置 PCINT13 位为“1”时，PC5 引脚电平改变中断被使能。PC5 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT13 位为“0”时，PC5 引脚电平改变中断被禁止。
4	PCINT12	引脚改变使能屏蔽位 12。 当设置 PCINT12 位为“1”时，PC4 引脚电平改变中断被使能。PC4 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT12 位为“0”时，PC4 引脚电平改变中断被禁止。
3	PCINT11	引脚改变使能屏蔽位 11。 当设置 PCINT11 位为“1”时，PC3 引脚电平改变中断被使能。PC3 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT11 位为“0”时，PC3 引脚电平改变中断被禁止。
2	PCINT10	引脚改变使能屏蔽位 2。 当设置 PCINT10 位为“1”时，PC2 引脚电平改变中断被使能。PC2 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT10 位为“0”时，PC2 引脚电平改变中断被禁

		止。
1	PCINT9	引脚改变使能屏蔽位 1。 当设置 PCINT9 位为“1”时，PC1 引脚电平改变中断被使能。PC1 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT9 位为“0”时，PC1 引脚电平改变中断被禁止。
0	PCINT8	引脚改变使能屏蔽位 0。 当设置 PCINT8 位为“1”时，PC0 引脚电平改变中断被使能。PC0 引脚上的电平改变将置位 PCIF1，若 PCIE1 位和全局中断置位，将会产生 PCI1 中断。当设置 PCINT8 位为“0”时，PC0 引脚电平改变中断被禁止。

PCMSK2 – 引脚改变中断屏蔽寄存器 2

PCMSK2 – 引脚改变屏蔽寄存器 2								
地址: 0x6D					默认值: 0x00			
Bits	7	6	5	4	3	2	1	0
	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7	PCINT23	引脚改变使能屏蔽位 23。 当设置 PCINT23 位为“1”时，PD7 引脚电平改变中断被使能。PD7 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT23 位为“0”时，PD7 引脚电平改变中断被禁止。						
6	PCINT22	引脚改变使能屏蔽位 6。 当设置 PCINT22 位为“1”时，PD6 引脚电平改变中断被使能。PD6 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT22 位为“0”时，PD6 引脚电平改变中断被禁止。						
5	PCINT21	引脚改变使能屏蔽位 21。 当设置 PCINT21 位为“1”时，PD5 引脚电平改变中断被使能。PD5 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT21 位为“0”时，PD5 引脚电平改变中断被禁止。						
4	PCINT20	引脚改变使能屏蔽位 20。 当设置 PCINT20 位为“1”时，PD4 引脚电平改变中断被使能。PD4 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT20 位为“0”时，PD4 引脚电平改变中断被禁止。						
3	PCINT19	引脚改变使能屏蔽位 19。 当设置 PCINT19 位为“1”时，PD3 引脚电平改变中断被使能。PD3 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT19 位为“0”时，PD3 引脚电平改变中断被禁止。						
2	PCINT18	引脚改变使能屏蔽位 18。						

		当设置 PCINT18 位为“1”时，PD2 引脚电平改变中断被使能。PD2 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT18 位为“0”时，PD2 引脚电平改变中断被禁止。
1	PCINT17	引脚改变使能屏蔽位 17。 当设置 PCINT17 位为“1”时，PD1 引脚电平改变中断被使能。PD1 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT17 位为“0”时，PD1 引脚电平改变中断被禁止。
0	PCINT16	引脚改变使能屏蔽位 16。 当设置 PCINT16 位为“1”时，PD0 引脚电平改变中断被使能。PD0 引脚上的电平改变将置位 PCIF2，若 PCIE2 位和全局中断置位，将会产生 PCI2 中断。 当设置 PCINT16 位为“0”时，PD0 引脚电平改变中断被禁止。

PCMSK3 – 引脚改变中断屏蔽寄存器 3

PCMSK3 – 引脚改变屏蔽寄存器 3								
地址: 0x73					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	PCINT31	PCINT30	PCINT29	PCINT28	PCINT27	PCINT26	PCINT25	PCINT24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7	PCINT31	引脚改变使能屏蔽位 31。 当设置 PCINT31 位为“1”时，PE7 引脚电平改变中断被使能。PE7 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT31 位为“0”时，PE7 引脚电平改变中断被禁止。						
6	PCINT30	引脚改变使能屏蔽位 30。 当设置 PCINT30 位为“1”时，PE6 引脚电平改变中断被使能。PE6 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT30 位为“0”时，PE6 引脚电平改变中断被禁止。						
5	PCINT29	引脚改变使能屏蔽位 39。 当设置 PCINT29 位为“1”时，PE5 引脚电平改变中断被使能。PE5 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT29 位为“0”时，PE5 引脚电平改变中断被禁止。						
4	PCINT28	引脚改变使能屏蔽位 28。 当设置 PCINT28 位为“1”时，PE4 引脚电平改变中断被使能。PE4 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT28 位为“0”时，PE4 引脚电平改变中断被禁止。						
3	PCINT27	引脚改变使能屏蔽位 27。						

		当设置 PCINT27 位为“1”时，PE3 引脚电平改变中断被使能。PE3 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT27 位为“0”时，PE3 引脚电平改变中断被禁止。
2	PCINT26	引脚改变使能屏蔽位 26。 当设置 PCINT26 位为“1”时，PE2 引脚电平改变中断被使能。PE2 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT26 位为“0”时，PE2 引脚电平改变中断被禁止。
1	PCINT25	引脚改变使能屏蔽位 25。 当设置 PCINT25 位为“1”时，PE1 引脚电平改变中断被使能。PE1 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT25 位为“0”时，PE1 引脚电平改变中断被禁止。
0	PCINT24	引脚改变使能屏蔽位 24。 当设置 PCINT24 位为“1”时，PE0 引脚电平改变中断被使能。PE0 引脚上的电平改变将置位 PCIF3，若 PCIE3 位和全局中断置位，将会产生 PCI3 中断。 当设置 PCINT24 位为“0”时，PE0 引脚电平改变中断被禁止。

PCMSK4 – 引脚改变中断屏蔽寄存器 4

PCMSK4 – 引脚改变屏蔽寄存器 4								
地址: 0x74					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	PCINT39	PCINT38	PCINT37	PCINT36	PCINT35	PCINT34	PCINT33	PCINT32
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7	PCINT39	引脚改变使能屏蔽位 39。 当设置 PCINT39 位为“1”时，PF7 引脚电平改变中断被使能。PF7 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT39 位为“0”时，PF7 引脚电平改变中断被禁止。						
6	PCINT38	引脚改变使能屏蔽位 38。 当设置 PCINT38 位为“1”时，PF6 引脚电平改变中断被使能。PF6 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT38 位为“0”时，PF6 引脚电平改变中断被禁止。						
5	PCINT37	引脚改变使能屏蔽位 37。 当设置 PCINT37 位为“1”时，PF5 引脚电平改变中断被使能。PF5 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT37 位为“0”时，PF5 引脚电平改变中断被禁止。						
4	PCINT36	引脚改变使能屏蔽位 36。						

		当设置 PCINT36 位为“1”时，PF4 引脚电平改变中断被使能。PF4 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT36 位为“0”时，PF4 引脚电平改变中断被禁止。
3	PCINT35	引脚改变使能屏蔽位 35。 当设置 PCINT35 位为“1”时，PF3 引脚电平改变中断被使能。PF3 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT35 位为“0”时，PF3 引脚电平改变中断被禁止。
2	PCINT34	引脚改变使能屏蔽位 34。 当设置 PCINT34 位为“1”时，PF2 引脚电平改变中断被使能。PF2 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT34 位为“0”时，PF2 引脚电平改变中断被禁止。
1	PCINT33	引脚改变使能屏蔽位 33。 当设置 PCINT33 位为“1”时，PF1 引脚电平改变中断被使能。PF1 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT33 位为“0”时，PF1 引脚电平改变中断被禁止。
0	PCINT32	引脚改变使能屏蔽位 32。 当设置 PCINT31 位为“1”时，PF0 引脚电平改变中断被使能。PF0 引脚上的电平改变将置位 PCIF4，若 PCIE4 位和全局中断置位，将会产生 PCI4 中断。 当设置 PCINT32 位为“0”时，PF0 引脚电平改变中断被禁止。

定时/计数器 0 (TMR0)

- 8 位计数器
- 两个独立的比较单元
- 比较匹配发生时自动清零计数器并自动加载
- 无干扰脉冲的相位修正的 PWM 输出
- 频率发生器
- 外部事件计数器
- 10 位的时钟预分频器
- 溢出和比较匹配中断
- 带死区时间控制
- 6 个可选触发源自动关闭 PWM 输出
- 高速时钟模式下产生高速高分辨率（500KHz@7Bit）PWM

概述

T0 是一个通用 8 位定时计数器模块，支持 PWM 输出，可以精确地产生波形。T0 包含 1 个计数时钟产生单元，1 个 8 位计数器，波形产生模式控制单元和 2 个输出比较单元。同时，T0 可与 T1 共用 10 位的预分频器，也可以独立使用 10 位的预分频器。预分频器对系统时钟 clkio 或高速时钟 rcm2x（内部 32M RC 振荡器输出时钟 rc32m 的 2 倍频）进行分频来产生计数时钟 Clkt0。波形产生模式控制单元控制着计数器的工作模式和比较输出波形的产生。根据不同的工作模式，计数器对每一个计数时钟 Clkt0 实现清零、加一或减一操作。Clkt0 可以由内部时钟源或外部时钟源产生。当计数器的计数值 TCNT0 到达最大值（等于极大值 0xFF 或输出比较寄存器 OCR0A，定义为 TOP，定义极大值为 MAX 以示区别）时，计数器会进行清零或减一操作。当计数器的计数值 TCNT0 到达最小值（等于 0x00，定义为 BOTTOM）时，计数器会进行加一操作。当计数器的计数值 TCNT0 到达 OCR0A/OCR0B 时，也被称为发生比较匹配时，会清零或置位输出比较信号 OC0A/OC0B，来产生 PWM 波形。当使能插入死区时间时，设定的死区时间（DTR0 寄存器所对应的计数时钟数）将会插入到已产生的 PWM 波形中。软件可通过清除 COM0A/COM0B 位为零来关闭 OC0A/OC0B 的波形输出，或者设置相应的触发源，当触发事件发生时硬件自动清零 COM0A/COM0B 位来关闭 OC0A/OC0B 的波形输出。

计数时钟可由内部或外部时钟源来产生，时钟源的选择及分频选择由位于 TCCR0B 寄存器的 CS0 位来控制，详细描述见 T0 和 T1 预分频器章节。

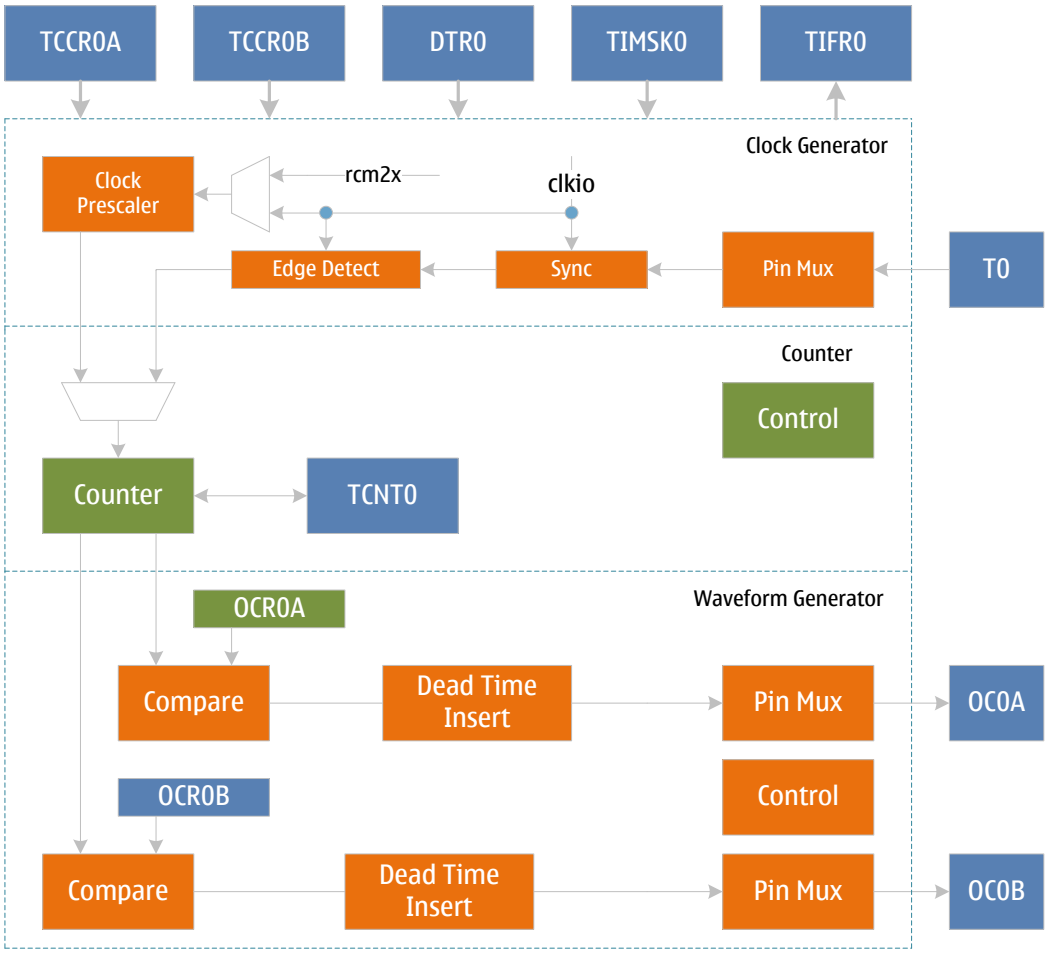
计数器的长度为 8 位，支持双向计数。波形产生模式即计数器的工作模式由位于 TCCR0A 和 TCCR0B 寄存器的 WGM0 位来控制。根据不同的工作模式，计数器对每一个计数时钟 Clkt0 实现清零、加一或减一操作。当计数发生溢出时，位于 TIFR0 寄存器的计数溢出标志 TOV0 位会被置位。当中断使能时可产生 T0 计数溢出中断。

输出比较单元对计数值 TCNT0 和输出比较寄存器 OCR0A 和 OCR0B 的值进行比较，当 TCNT0 等于 OCR0A 或 OCR0B 时称为发生比较匹配，位于 TIFR0 寄存器的输出比较标志 OCF0A 或 OCF0B 位会被置位。当中断使能时可产生 T0 输出比较匹配中断。
需注意的是，在 PWM 工作模式下，OCR0A 和 OCR0B 寄存器为双缓冲寄存器。在普通模式和

CTC 模式下，双缓冲功能失效。计数到达最大值或最小值时，缓冲寄存器中的值被同步更新到比较寄存器 OCR0A 和 OCR0B 中去。详见工作模式章节描述。

波形产生器根据波形产生模式控制和比较输出模式控制使用比较匹配和计数溢出等来产生输出比较波形信号 OC0A 和 OC0B。具体产生方式见工作模式和寄存器章节描述。要把输出比较波形信号 OC0A 和 OC0B 输出到相应引脚上时，还必须设置该引脚的数据方向寄存器为输出。

下图为 TC0 的内部结构图。TC0 包含 1 个计数时钟产生单元，1 个 8 位计数器，2 个输出比较单元和 2 个波形产生控制单元。



TC0 结构图

工作模式

定时计数器 0 有四种不同的工作模式，包括普通模式（Normal），比较匹配时清零（CTC）模式，快速脉冲宽度调制（FPWM）模式和相位修正脉冲宽度调制（PCPWM）模式，由波形产生模式控制位 WGM0[2:0]来选择。下面具体来描述这四种模式。由于有两个独立的输出比较单元，分别用“A”和“B”来表示，用小写的“x”来表示这两个输出比较单元通道。

普通模式

普通模式是定时计数器最简单的工作模式，此时波形产生模式控制位 WGM0[2:0]=0，计数的最大值 TOP 为 MAX（0xFF）。在这种模式下，计数方式为每一个计数时钟加一递增，当计数器到达 TOP 溢出后就回到 BOTTOM 重新开始累加。在计数值 TCNT0 变成零的同一个计数时钟里置位定时计数器溢出标志 TOV0。这种模式下 TOV0 标志就像是第 9 计数位，只是只会被置位不会被清零。溢出中断服务程序会自动清除 TOV0 标志，软件可以用它来提高定时计数器的分辨率。普通模式下没有特殊情形需要考虑，可以随时写入新的计数值。

设置 OC0x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC0x 的波形。当 COM0x=1 时，发生比较匹配时会翻转 OC0x 信号，这种情况下波形的频率可以用下面的公式来计算：

$f_{oc0xnormal} = f_{sys}/(2*N*256)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

输出比较单元可以用来产生中断，但是在普通模式下不推荐使用中断，这样会占用太多 CPU 的时间。

CTC 模式

设置 WGM0[2:0]=2 时，定时计数器 0 进入 CTC 模式，计数的最大值 TOP 为 OCR0A。在这个模式下，计数方式为每一个计数时钟加一递增，当计数器的数值 TCNT0 等于 TOP 时计数器清零。OCR0A 定义了计数的最大值，亦即计数器的分辨率。这个模式使得用户可以很容易的控制比较匹配输出的频率，也简化了外部事件计数的操作。

当计数器到达计数的最大值时，输出比较匹配标志 OCF0 被置位，相应的中断使能置位时将会产生中断。在中断服务程序里可以更新 OCR0A 寄存器即计数的最大值。在这个模式下 OCR0A 没有使用双缓冲，在计数器以无预分频器或很低的预分频器工作下将最大值更新为接近最小值的时候要小心。如果写入 OCR0A 的数值小于当时的 TCNT0 值时，计数器将丢失一次比较匹配。在下一次比较匹配发生之前，计数器不得不先计数到 TOP，然后再从 BOTTOM 开始计数到 OCR0A 值。和普通模式一样，计数值回到 BOTTOM 的计数时钟里置位 TOV0 标志。

设置 OC0x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC0x 的波形。当 COM0x=1 时，发生比较匹配时会翻转 OC0x 信号，这种情况下波形的频率可以用下面的公式来计算：

$f_{oc0xctc} = f_{sys}/(2*N*(1+OCR0x))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

从公式可以看出，当设置 OCR0A 为 0x0 且无预分频器时，可以获得最大频率为 $f_{sys}/2$ 的输出波形。

快速 PWM 模式

设置 WGM0[2:0]=3 或 7 时，定时计数器 0 进入快速 PWM 模式，可以用来产生高频的 PWM 波形，计数最大值 TOP 分别为 MAX（0xFF）或 OCR0x。快速 PWM 模式和其他 PWM 模式不同在于它是单向操作。计数器从最小值 0x00 累加到 TOP 后又回到 BOTTOM 重新计数。当计数值 TCNT0 到达 OCR0x 或 BOTTOM 时，输出比较信号 OC0x 会被置位或清零，取决于比较输出模式 COM0x 的设置，详情见寄存器描述。由于采用单向操作，快速 PWM 模式的操作频率是采用双向操作的相位修正 PWM 模式的两倍。高频特性使得快速 PWM 模式适用于功率调节，整流以及 DAC 应用。高频信号可以减小外部元器件（电感电容等）的尺寸，从而降低系统成本。

当计数值到达最大值时，定时计数器溢出标志 TOV0 将会被置位，并把比较缓冲器的值更新

到比较值。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR0x 寄存器。
设置 OC0x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC0x 的波形。波形的频率可用下面的公式来计算：

$f_{oc0xfpwm} = f_{sys}/(N*(1+TOP))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。
当 TCNT0 和 OCR0x 发生比较匹配时，波形产生器就置位（清零）OC0x 信号，当 TCNT0 被清零时，波形产生器就清零（置位）OC0x 信号，以此来产生 PWM 波。由此 OCR0x 的极值将会产生特殊的 PWM 波形。当 OCR0x 设置为 0x00 时，输出的 PWM 为每(1+TOP)个计数时钟里有一个窄的尖峰脉冲。当 OCR0x 设置为最大值时，输出的波形为持续的高电平或低电平。

相位修正 PWM 模式

当设置 WGM0[2:0]=1 或 5 时，定时计数器 0 进入相位修正 PWM 模式，计数的最大值 TOP 分别为 MAX（0xFF）或 OCR0A。计数器采用双向操作，由 BOTTOM 递增到 TOP，然后又递减到 BOTTOM，再重复此操作。计数到达 TOP 和 BOTTOM 时均改变计数方向，计数值在 TOP 或 BOTTOM 上均只停留一个计数时钟。在递增或递减过程中，计数值 TCNT0 与 OCR0x 匹配时，输出比较信号 OC0x 将会被清零或置位，取决于比较输出模式 COM0x 的设置。与单向操作相比，双向操作可获得的最大频率要小，但其极好的对称性更适合于电机控制。
相位修正 PWM 模式下，当计数到达 BOTTOM 时置位 TOV0 标志，当计数到达 TOP 时把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR0x 寄存器。
设置 OC0x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC0x 的波形。波形的频率可用下面的公式来计算：

$f_{oc0xpcpwm} = f_{sys}/(N*TOP*2)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。
在递增计数过程中，当 TCNT0 与 OCR0x 匹配时，波形产生器就清零（置位）OC0x 信号。在递减计数过程中，当 TCNT0 与 OCR0x 匹配时，波形产生器就置位（清零）OC0x 信号。由此 OCR0x 的极值会产生特殊的 PWM 波。当 OCR0x 设置为最大值或最小值时，OC0x 信号输出会一直保持低电平或高电平。

为了保证输出 PWM 波在最小值两侧的对称性，在没有发生比较匹配时，有两种情况下也会翻转 OC0x 信号。第一种情况是，当 OCR0x 的值由最大值 0xFF 改变为其他数据时。当 OCR0x 为最大值，计数值达到最大时，OC0x 的输出与前面降序计数时比较匹配的结果相同，即保持 OC0x 不变。此时会更新比较值为新的 OCR0x 的值（非 0xFF），OC0x 的值会一直保持，直到升序计数时发生比较匹配而翻转。此时 OC0x 信号并不以最小值为中心对称，因此需要在 TCNT0 到达最大值时翻转 OC0x 信号，此即没有发生比较匹配时翻转 OC0x 信号的第一种情况。第二种情况是，当 TCNT0 从比 OCR0x 高的值开始计数时，因而会丢失一次比较匹配，从而引起不对称情形的产生。同样需要翻转 OC0x 信号去实现最小值两侧的对称性。

PWM 输出的自动关闭与重启

当设置 TCCR0A 寄存器的 DOC0x 位为高时，PWM 输出的自动关闭功能会被使能，满足触发条件时，硬件会清零相应的 COM0x 位，将 PWM 输出信号 OC0x 与其输出引脚断开，切换成通用 IO 输出，实现 PWM 输出的自动关闭。此时，输出引脚的状态可由通用 IO 口的输出来控制。

PWM 输出的自动关闭被使能后，还需要设置其触发条件，由 TCCR0C 寄存器的 DSX0n 位来选择触发源。触发源有模拟比较器中断，外部中断，引脚电平变化中断以及定时器溢出中断，具体情形请参考 TCCR0C 寄存器描述。当某个或某些触发源被选用作为触发条件后，在这些中断标志位被置位的同时，硬件会清零 COM0x 位来关闭 PWM 的输出。

当发生了触发事件关闭 PWM 输出后，定时器模块没有相应的中断标志位，软件需要通过读取触发源的中断标志位来得知触发条件和触发事件。

当 PWM 输出被自动关闭而需要再次重启输出时，软件只需要重新设置 COM0x 位，来切换 OC0x 信号输出到相应的引脚上。需要注意的是，发生自动关闭后，定时器并未停止工作，OC0x 信号的状态也一直在更新。软件可在定时器发生溢出或比较匹配后，再设置 COM0x 位来输出 OC0x 信号，这样可以获得明确的 PWM 输出状态。

死区时间控制

设置 DTEN0 位为“1”时，插入死区时间的功能被使能，OC0A 和 OC0B 的输出波形将在 B 通道比较输出所产生的波形基础上插入设定的死区时间，时间的长度为 DTR0 寄存器的计数时钟数所对应的时间值。如下图所示，OC0A 和 OC0B 的死区时间插入均是以通道 B 的比较输出波形为基准。当 COM0A 和 COM0B 同为“2”或“3”时，OC0A 的波形极性与 OC0B 的波形极性相同，当 COM0A 和 COM0B 分别为“2”或“3”时，OC0A 的波形与 OC0B 的波形极性相反。

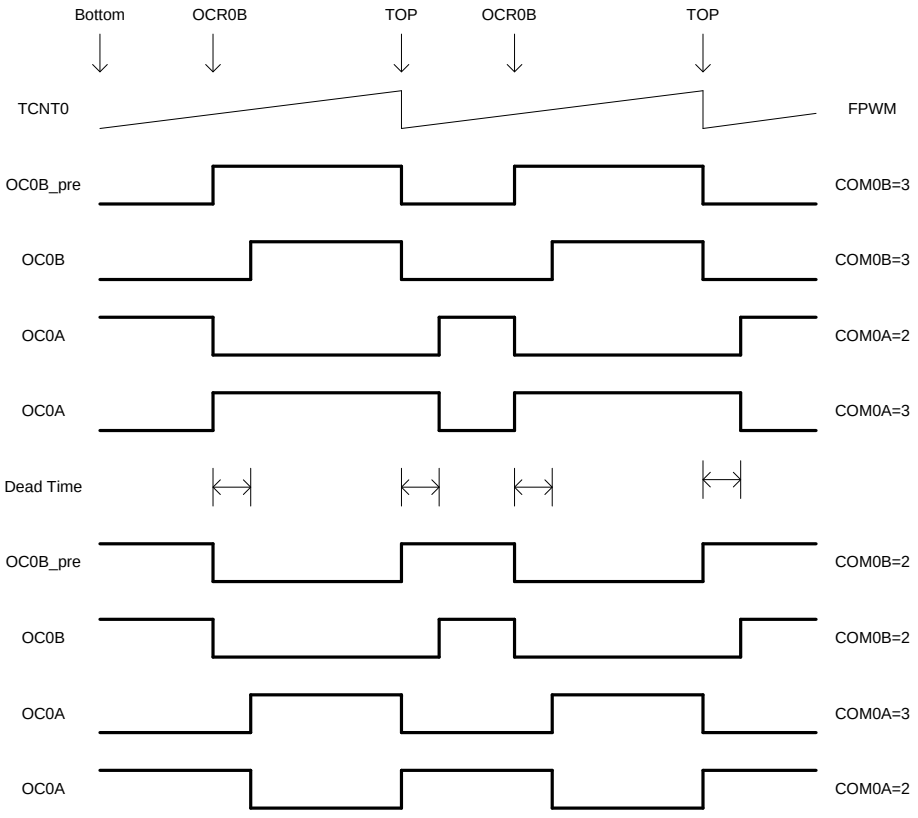


Figure 1 FPWM 模式下 TC0 死区时间控制

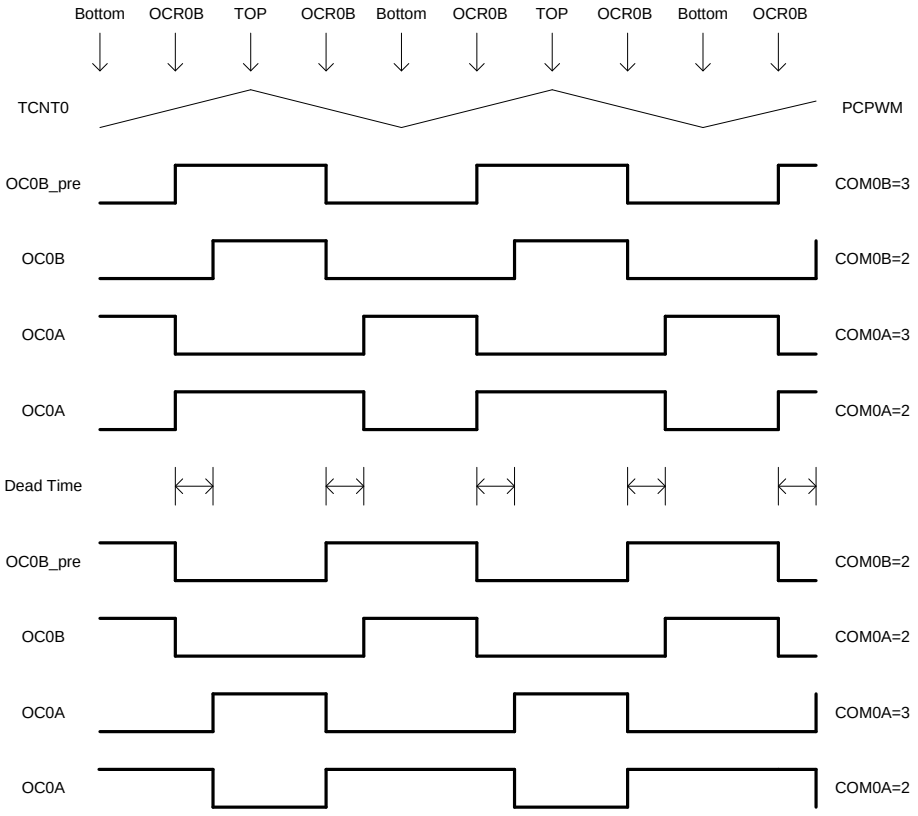


Figure 2 PCPWM 模式下 TC0 死区时间控制

设置 DTEN0 位为“0”时，插入死区时间的功能被禁止，OC0A 和 OC0B 的输出波形为各自比较输出所产生的波形。

高速时钟模式

高速时钟模式下，采用更高频率的时钟作为计数的时钟源，用来产生更高速度和更高分辨率的 PWM 波形。此高频时钟是通过对内部 32M RC 振荡器的输出时钟 rc32m 进行 2 倍频来产生的。因此，在进入高频模式之前，需先使能内部 32M RC 振荡器的倍频功能，即置位 TCKCSR 寄存器的 F2XEN 位，并等待一定时间直到倍频时钟信号输出稳定。然后，可置位 TCKCSR 的 TC2XS0 位来使定时计数器进入高速时钟模式。

在此模式下，系统时钟与高速时钟是异步关系，而部分寄存器（见 TC0 寄存器列表）是工作在高速时钟域，因此，配置和读取这类寄存器时也是异步的，操作时需注意。

对高速时钟域下的寄存器进行非连续读写操作时无特殊要求，而进行连续读写操作时，需等待一个系统时钟，可按以下步骤：

1） 写寄存器 A；
2） 等待一个系统时钟（NOP 或操作系统时钟下的寄存器）；
3） 读或写寄存器 A 或 B。
4） 等待一个系统时钟（NOP 或操作系统时钟下的寄存器）。

对高速时钟域下的寄存器进行读操作时，除 TCNT0 外的寄存器均可直接读取，当计数器还在进行计数时，TCNT0 的值会随高速时钟变化，可暂停计数器（设置 CS0 为零）再读取 TCNT0 的值。

寄存器定义

T0 寄存器列表

寄存器	地址	默认值	描述
TCCR0A*	0x44	0x00	T0 控制寄存器 A
TCCR0B*	0x45	0x00	T0 控制寄存器 B
TCNT0*	0x46	0x00	T0 计数值寄存器
OCR0A*	0x47	0x00	T0 输出比较寄存器 A
OCR0B*	0x48	0x00	T0 输出比较寄存器 B
DSX0*	0x49	0x00	T0 触发源控制寄存器
DTR0*	0x4F	0x00	T0 死区时间寄存器
TIMSK0	0x6E	0x00	定时计数器 0 中断屏蔽寄存器
TIFR0	0x35	0x00	定时计数器 0 中断标志寄存器
TCKCSR	0xEC	0x00	TC 时钟控制和状态寄存器

【注意】
带“*”的寄存器工作于系统时钟和高速时钟域下，未带“*”的寄存器仅工作于系统时钟域下。

T0 控制寄存器 A- TCCR0A

TCCR0A –T0 控制寄存器 A								
地址: 0x44							默认值: 0x00	
Bit	7	6	5	4	3	2	1	0
	COM0A1	COM0A0	COM0B1	COM0B0	DOC0B	DOC0A	WGM01	WGM00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	COM0A1	T0 比较匹配 A 输出模式控制高位。 COM0A1 和 COM0A0 一起组成比较输出模式控制 COM0A[1:0]，用来控制 OC0A 的输出波形。如果 COM0A 的 1 位或者 2 位都置位，输出比较波形占据着 OC0A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM0A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
6	COM0A0	T0 比较匹配 A 输出模式控制低位。 COM0A0 和 COM0A1 一起组成比较输出模式控制 COM0A[1:0] ，用来控制 OC0A 的输出波形。如果 COM0A 的 1 位或者 2 位都置位，输出比较波形占据着 OC0A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM0A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
5	COM0B1	T0 比较匹配 B 输出模式控制高位。 COM0B1 和 COM0B0 一起组成比较输出模式控制 COM0B[1:0]，用来控制 OC0B 的输出波形。如果 COM0B 的 1 位或者 2 位都置位，

		输出比较波形占据着 OC0B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM0B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
4	COM0B0	T0 比较匹配 B 输出模式控制低位。 COM0B0 和 COM0B1 一起组成比较输出模式控制 COM0B[1:0] ，用来控制 OC0B 的输出波形。如果 COM0B 的 1 位或者 2 位都置位，输出比较波形占据着 OC0B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM0B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
3	DOC0B	T0 关闭输出比较使能控制高位。 当 DOC0B 位为“1”时，触发源关闭输出比较信号 OC0B 被使能。当发生触发事件时，硬件自动清零 COM0B 位，关闭 OC0B 的波形输出。软件通过设置 COMB 可重新开启 PWM 输出。 当 DOC0B 位为“0”时，触发源关闭输出比较信号 OC0B 被禁止。
2	DOC0A	T0 关闭输出比较使能控制低位。 当设置 DOC0A 位为“1”时，触发源关闭输出比较信号 OC0A 被使能。当发生触发事件时，硬件自动关闭 OC0A 的波形输出。 当设置 DOC0A 位为“0”时，触发源关闭输出比较信号 OC0A 被禁止。当发生触发事件时，不会关闭 OC0A 的波形输出。
1	WGM01	T0 波形产生模式控制中位。 WGM01 和 WGM00，WGM02 一起组成波形产生模式控制 WGM0[2:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
0	WGM00	T0 波形产生模式控制低位。 WGM00 和 WGM01，WGM02 一起组成波形产生模式控制 WGM0[2:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。

T0 控制寄存器 B- TCCR0B

TCCR0B –T0 控制寄存器 B								
地址: 0x45					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	FOC0A	FOC0B	OC0AS	DTEN0	WGM02	CS02	CS01	CS00
R/W	W	W	W/R	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	FOC0A	T0 强制输出比较 A 控制位。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC0A 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF0A 标志，也不会重载或清零定时器，但是输出引脚 OC0A 将被按照 COM0A 的设置相应的更新，就跟真的发生了比较匹配一样。 读取 FOC0A 的返回值一直为零。
6	FOC0B	T0 强制输出比较 B 控制位。

<table>
<tr><td></td><td></td><td>工作于非 PWM 模式时，可以通过对强制输出比较位 FOC0B 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF0B 标志，也不会重载或清零定时器，但是输出引脚 OC0B 将被按照 COM0B 的设置相应的更新，就跟真的发生了比较匹配一样。
读取 FOC0B 的返回值一直为零。</td></tr>
<tr><td>5</td><td>OC0AS</td><td>OC0A 输出端口选择控制位。当设置 OC0AS 位为“0”时，OC0A 的波形从引脚 PD6 输出；当设置 OC0AS 位为“1”时，OC0A 的波形从引脚 PE4 输出（QFP32 封装下有效）。</td></tr>
<tr><td>4</td><td>DTEN0</td><td>T0 死区时间使能控制位。
当设置 DTEN0 位为“1”时，使能死区时间插入。OC0A 和 OC0B 均在 B 通道比较输出产生的波形基础上插入死区时间，所插入的死区时间间隔由 DTR0 寄存器所对应的计数时间决定。OC0A 输出波形的极性由 COM0 和 COM0B 的对应关系决定，详见 OC0A 插入死区时间后波形极性表格所示。
当设置 DTEN0 位为“0”时，禁止死区时间插入，OC0A 和 OC0B 的波形为各自比较输出所产生的波形。</td></tr>
<tr><td>3</td><td>WGM02</td><td>T0 波形产生模式控制高位。
WGM02 和 WGM00，WGM01 一起组成波形产生模式控制 WGM0[2:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。</td></tr>
<tr><td>2</td><td>CS02</td><td>T0 时钟选择控制高位。
用于选择定时计数器 0 的时钟源。</td></tr>
<tr><td>1</td><td>CS01</td><td>T0 时钟选择控制中位。
用于选择定时计数器 0 的时钟源。</td></tr>
<tr><td rowspan="10">0</td><td rowspan="10">CS00</td><td>T0 时钟选择控制低位。
用于选择定时计数器 0 的时钟源。</td></tr>
<tr><td><table>
<tr><th>CS0[2:0]</th><th>描述</th></tr>
<tr><td>0</td><td>无时钟源，停止计数</td></tr>
<tr><td>1</td><td>clk_{sys}</td></tr>
<tr><td>2</td><td>clk_{sys}/8，来自预分频器</td></tr>
<tr><td>3</td><td>clk_{sys}/64，来自预分频器</td></tr>
<tr><td>4</td><td>clk_{sys}/256，来自预分频器</td></tr>
<tr><td>5</td><td>clk_{sys}/1024，来自预分频器</td></tr>
<tr><td>6</td><td>外部时钟 T0 引脚，下降沿触发</td></tr>
<tr><td>7</td><td>外部时钟 T0 引脚，上升沿触发</td></tr>
</table></td></tr>
</table>

下表为非 PWM 模式（即普通模式和 CTC 模式）下，比较输出模式对输出比较波形的控制。

COM0x[1:0]	描述
0	OC0x 断开，通用 IO 口操作
1	比较匹配时翻转 OC0x 信号
2	比较匹配时清零 OC0x 信号
3	比较匹配时置位 OC0x 信号

下表为快速 PWM 模式下比较输出模式对输出比较波形的控制。

COM0x[1:0]	描述
0	OC0x 断开，通用 IO 口操作
1	保留
2	比较匹配时清零 OC0x 信号，最大值匹配时置位 OC0x 信号
3	比较匹配时置位 OC0x 信号，最大值匹配时清零 OC0x 信号

下表为相位修正模式下比较输出模式对输出比较波形的控制。

COM0x[1:0]	描述
0	OC0x 断开，通用 IO 口操作
1	保留
2	升序计数下比较匹配时清零 OC0x 信号，降序计数下比较匹配时置位 OC0x 信号
3	升序计数下比较匹配时置位 OC0x 信号，降序计数下比较匹配时清零 OC0x 信号

下表为波形产生模式控制。

WGM0[2:0]	工作模式	TOP 值	更新 OCR0X 时刻	置位 TOV0 时刻
0	Normal	0xFF	立即	MAX
1	PCPWM	0xFF	TOP	BOTTOM
2	CTC	OCR0A	立即	MAX
3	FPWM	0xFF	TOP	MAX
4	保留	-	-	-
5	PCPWM	OCR0A	TOP	BOTTOM
6	保留	-	-	-
7	FPWM	OCR0A	TOP	TOP

下表为死区时间使能时 OC0A 信号输出波形的极性控制。

死区时间使能模式下 OC0A 信号输出波形的极性控制

DTEN0	COM0A[1:0]	COM0B[1:0]	描述
0	-	-	OC0A 信号极性由 OC0A 比较输出模式控制
1	0	-	OC0A 断开，通用 IO 口操作
1	1	-	保留
1	2	2	OC0A 信号与 OC0B 信号极性相同
		3	OC0A 信号与 OC0B 信号极性相反
1	3	2	OC0A 信号与 OC0B 信号极性相反
		3	OC0A 信号与 OC0B 信号极性相同

【注意】:

OC0B 信号输出波形的极性由 OC0B 比较输出模式控制，与未使能死区时间模式相同。

TC0 控制寄存器 C – TCCR0C

TCCR0C - *TC0 控制寄存器 C*								
地址: 0x49					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	DSX07	DSX06	DSX05	DSX04	-	-	DSX01	DSX00
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	描述
7	DSX07	TC0 触发源选择控制使能第 7 位。 当设置 DSX07 位为“1”时，TC1 溢出作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被使能。当 DOC0A/DOC0B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC0A/OC0B 的波形输出。 当设置 DSX07 位为“0”时，TC1 溢出作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被禁止。
6	DSX06	TC0 触发源选择控制使能第 6 位。 当设置 DSX06 位为“1”时，TC2 溢出作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被使能。当 DOC0A/DOC0B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC0A/OC0B 的波形输出。 当设置 DSX06 位为“0”时，TC2 溢出作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被禁止。
5	DSX05	TC0 触发源选择控制使能第 5 位。 当设置 DSX05 位为“1”时，引脚电平变化 0 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被使能。当 DOC0A/DOC0B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC0A/OC0B 的波形输出。 当设置 DSX05 位为“0”时，引脚电平变化 0 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被禁止。
4	DSX04	TC0 触发源选择控制使能第 4 位。 当设置 DSX04 位为“1”时，外部中断 0 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被使能。当 DOC0A/DOC0B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC0A/OC0B 的波形输出。 当设置 DSX04 位为“0”时，外部中断 0 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被禁止。
3:2	-	保留不用
1	DSX01	TC0 触发源选择控制使能第 1 位。 当设置 DSX01 位为“1”时，模拟比较器 1 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被使能。当 DOC0A/DOC0B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC0A/OC0B 的波形输出。

		当设置 DSX01 位为“0”时，模拟比较器 1 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被禁止。
0	DSX00	TC0 触发源选择控制使能第 0 位。 当设置 DSX00 位为“1”时，模拟比较器 0 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被使能。当 DOC0A/DOC0B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC0A/OC0B 的波形输出。 当设置 DSX00 位为“0”时，模拟比较器 0 作为为关闭输出比较信号波形 OC0A/OC0B 的触发源被禁止。

下表为波形输出的触发源的选择控制。

关闭 OC0A/OC0B 波形输出的触发源选择控制

DOC0x	DSX0n=1	触发源	描述
0	-	-	DOC0x 位为“0”，触发源关闭波形输出功能被禁止
1	0	模拟比较器 0	ACIF0 的上升沿将关闭 OC0x 波形输出
1	1	模拟比较器 1	ACIF1 的上升沿将关闭 OC0x 波形输出
1	4	外部中断 0	INTF0 的上升沿将关闭 OC0x 波形输出
1	5	引脚电平变化 0	PCIF0 的上升沿将关闭 OC0x 波形输出
1	6	TC2 溢出	TOV2 的上升沿将关闭 OC0x 波形输出
1	7	TC1 溢出	TOV1 的上升沿将关闭 OC0x 波形输出

注意：

1） DSX0n=1 表示 DSX0 寄存器的第 n 位为 1 时，各寄存器位可同时置位。

TC0 计数值寄存器- TCNT0

TCNT0 –*TC0 计数值寄存器*								
地址: 0x46					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT07	TCNT06	TCNT05	TCNT04	TCNT03	TCNT02	TCNT01	TCNT00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	TCNT0	TC0 计数值寄存器。 通过 TCNT0 寄存器可以直接对计数器的 8 为计数值进行读写访问。CPU 对 TCNT0 寄存器的写操作会在下一个定时器时钟周期阻止比较匹配的发生，即使定时器已经停止。这就允许初始化 TCNT0 寄存器的值与 OCR0 的值一致而不会引发中断。 如果写入 TCNT0 的数值等于或绕过 OCR0 值时，比较匹配就会丢失，造成不正确的波形发生结果。 没有选择时钟源时定时器停止计数，但 CPU 仍可以访问 TCNT0。CPU 写计数器比清零或加减操作的优先级高。

T0 输出比较寄存器 A- OCR0A

OCR0A – *T0 输出比较寄存器 A*								
地址: 0x47					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR0A7	OCR0A6	OCR0A5	OCR0A4	OCR0A3	OCR0A2	OCR0A1	OCR0A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	OCR0A	T0 输出比较寄存器。 OCR0A 包含一个 8 位的数据，不间断地与计数器数值 TCNT0 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC0A 引脚上产生波形。 当使用 PWM 模式时，OCR0A 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR0A 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR0A 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR0A 本身。

T0 输出比较寄存器 B- OCR0B

OCR0B – T0 输出比较寄存器 B								
地址: 0x48					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR0B7	OCR0B6	OCR0B5	OCR0B4	OCR0B3	OCR0B2	OCR0B1	OCR0B0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	描述
7:0	OCR0B	T0 输出比较 B 寄存器。 OCR0B 包含一个 8 位的数据，不间断地与计数器数值 TCNT0 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC0B 引脚上产生波形。 当使用 PWM 模式时，OCR0B 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR0B 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR0B 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR0B 本身。

T0 中断屏蔽寄存器- TIMSK0

TIMSK0 – T0 中断屏蔽寄存器								
地址: 0x6E					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0
R/W	-	-	-	-	-	R/W	R/W	R/W

Bit	Name	描述
7:3		保留。
2	OCIE0B	T0 输出比较 B 匹配中断使能位。 当 OCIE0B 位为“1”，且全局中断置位，T0 输出比较 B 匹配中断使能。当比较匹配发生时，即 TIFR0 中 OCF0B 位被置位时，中断产生。 当 OCIE0B 位为“0”时，T0 输出比较 B 匹配中断被禁止。
1	OCIE0A	T0 输出比较 A 匹配中断使能位。 当 OCIE0A 位为“1”，且全局中断置位，T0 输出比较 A 匹配中断使能。当比较匹配发生时，即 TIFR0 中 OCF0A 位被置位时，中断产生。 当 OCIE0A 位为“0”时，T0 输出比较 A 匹配中断被禁止。
0	TOIE0	T0 溢出中断使能位。 当 TOIE0 位为“1”，且全局中断置位，T0 溢出中断使能。当 T0 发生溢出，即 TIFR 中的 TOV0 位被置位时，中断产生。 当 TOIE0 位为“0”时，T0 溢出中断被禁止。

T0 中断标志寄存器- TIFR0

TIFR0 – T0 中断标志寄存器								
地址: 0x35					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OC0A	OC0B	-	-	-	OCF0B	OCF0A	TOV0
R/W	R/O	R/O	-	-	-	R/W	R/W	R/W

Bit	Name	描述
7	OC0A	输出比较波形信号 OC0A。 输出比较波形信号 OC0A，软件可读但不可写。软件可在未使能 OC0A 信号输出至其相应 IO 引脚上之前，可先读取 OC0A 位的值来获取将要输出比较波形信号的极性，并可通过配置 COM0A 位和置位 FOC0A 位来改变其极性，避免在使能 OC0A 信号输出至其相应 IO 引脚上之后产生多余的干扰脉冲。
6	OC0B	输出比较波形信号 OC0B。 输出比较波形信号 OC0B，软件可读但不可写。软件可在未使能 OC0B 信号输出至其相应 IO 引脚上之前，可先读取 OC0B 位的

		值来获取将要输出比较波形信号的极性，并可通过配置 COM0B 位和置位 FOC0B 位来改变其极性，避免在使能 OC0B 信号输出至其相应 IO 引脚上之后产生多余的干扰脉冲。
5:3		保留
2	OCF0B	TC0 输出比较 B 匹配标志位。 当 TCNT0 等于 OCR0B 时，比较单元就给出匹配信号，并置位比较标志 OCF0B。若此时输出比较 B 中断使能 OCIE0B 为“1”且全局中断标志置位，则会产生输出比较 B 中断。执行此中断服务程序时 OCF0B 将自动清零，或对 OCF0B 位写“1”也可清零该位。
1	OCF0A	TC0 输出比较 A 匹配标志位。 当 TCNT0 等于 OCR0A 时，比较单元就给出匹配信号，并置位比较标志 OCF0A。若此时输出比较 A 中断使能 OCIE0A 为“1”且全局中断标志置位，则会产生输出比较 A 中断。执行此中断服务程序时 OCF0A 将自动清零，或对 OCF0A 位写“1”也可清零该位。
0	TOV0	TC0 溢出标志位。 当计数器发生溢出时，置位溢出标志 TOV0。若此时溢出中断使能 TOIE0 为“1”且全局中断标志置位，则会产生溢出中断。执行此中断服务程序时 TOV0 将自动清零，或对 TOV0 位写“1”也可清零该位。

DTR0 - TC0 死区时间控制寄存器

DTR0 – TC0 死区时间控制寄存器								
地址: 0x4F					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	DTR07	DTR06	DTR05	DTR04	DTR03	DTR02	DTR01	DTR00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
[7:4]	DTR0H	TC0 死区时间寄存器高位。 当 TCCR0B 寄存器的 DTEN0 位为“1”时，OC0A 和 OC0B 组成互补输出，插入死区时间控制被使能，OC0B 通道上所插入的死区时间由 DTR0H 决定，时间的长度为 DTR0H 个计数时钟所对应的时间。
[3:0]	DTR0L	TC0 死区时间寄存器低位。 当 TCCR0B 寄存器的 DTEN0 位为“1”时，OC0A 和 OC0B 组成互补输出，插入死区时间控制被使能，OC0A 通道上所插入的死区时间由 DTR0L 决定，时间的长度为 DTR0H 个计数时钟所对应的时间。

TCKCSR - TC 时钟控制与状态寄存器

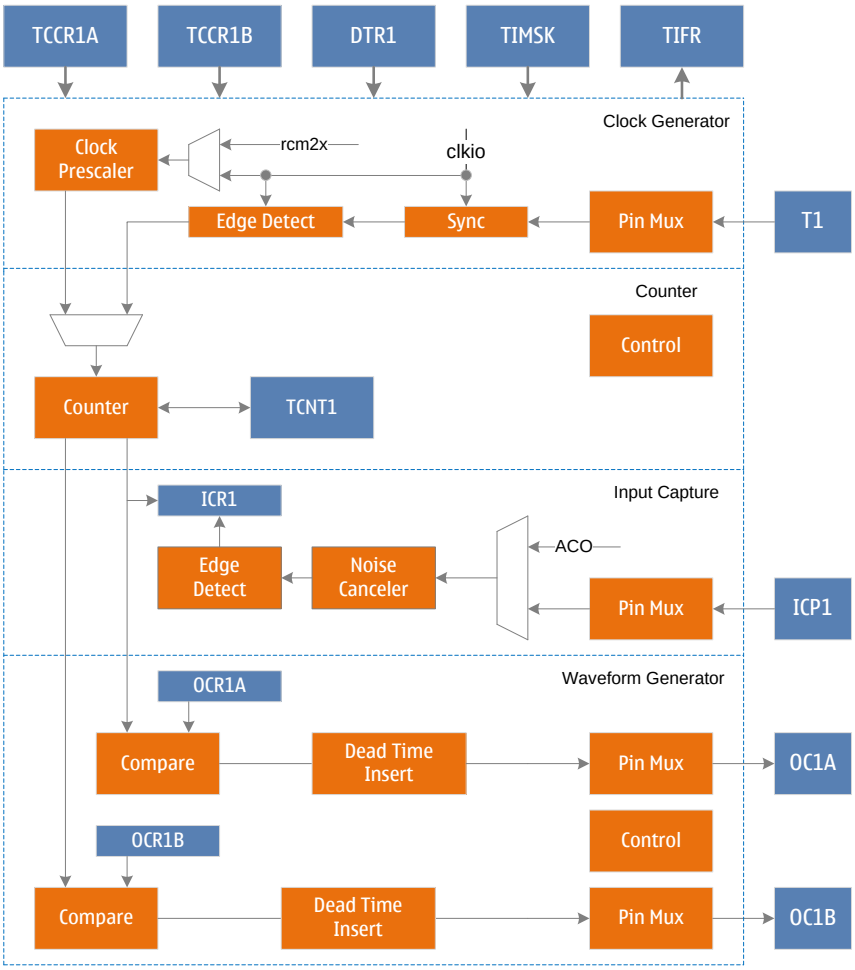
TCKSCR – TC 时钟控制与状态寄存器								
地址: 0xEC					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	F2XEN	TC2XF1	TC2XF0	-	AFCKS	TC2XS1	TC2XS0
R/W	-	R/W	R	R	-	R/W	R/W	R/W

Bit	Name	描述
7	-	保留
6	F2XEN	RC 32M 倍频输出使能控制位。 当设置 F2XEN 位为“1”时，32M RC 振荡器的倍频输出被使能，输出 64M 的高速时钟。 当设置 F2XEN 位为“1”时，32M RC 振荡器的倍频输出被禁止，不能输出 64M 的高速时钟。
5	TC2XF1	TC 高速时钟模式标志位 1。 见定时计数器 1 寄存器描述。
4	TC2XF0	TC 高速时钟模式标志位 0。 当读到 TC2XF0 位为“1”时，表明定时计数器 0 工作于高速时钟模式，为“0”时，表明定时计数器 0 工作于系统时钟模式。
3:2	-	保留。
1	TC2XS1	TC 高速时钟模式选择控制位 1。 见定时计数器 1 寄存器描述。
0	TC2XS0	TC 高速时钟模式选择控制位 0。 当设置 TC2XS0 位为“1”时，选择定时计数器 0 工作于高速时钟模式。 当设置 TC2XS0 位为“0”时，选择定时计数器 0 工作于系统时钟模式。

定时/计数器 1 (TMR1)

- 真正的 16 位设计，允许 16 位的 PWM
- 2 个独立的输出比较单元
- 双缓冲的输出比较寄存器
- 1 个输入捕捉单元
- 输入捕捉噪声抑制器
- 比较匹配时自动清零计数器并自动加载
- 无干扰脉冲的相位修正的 PWM
- 可变的 PWM 周期
- 频率发生器
- 外部事件计数器
- 4 个独立的中断源
- 支持死区时间控制的 PWM
- 6 个可选触发源自动关闭 PWM 输出
- 高速时钟模式下产生高速高分辨率(500KHZ@7BIT) PWM

概述



TC1 结构图

TC1 是一个通用 16 位定时计数器模块，支持 PWM 输出，可以精确地产生波形。TC1 包含 1 个 16 位计数器，波形产生模式控制单元，2 个独立的输出比较单元和 1 个输入捕捉单元。同时，TC1 可与 TC0 共用 10 位的预分频器，也可以独立使用 10 位的预分频器。预分频器对系统时钟 clkio 或高速时钟 rcm2x（内部 32M RC 振荡器输出时钟 rc32m 的 2 倍频）进行分频来产生计数时钟 Clkt1。波形产生模式控制单元控制着计数器的工作模式和比较输出波形的产生。根据不同的工作模式，计数器对每一个计数时钟 Clkt1 实现清零、加一或减一操作。Clkt1 可以由内部时钟源或外部时钟源产生。当计数器的计数值 TCNT1 到达最大值（等于极大值 0xFFFF 或固定值或输出比较寄存器 OCR1A 或输入捕捉寄存器 ICR1，定义为 TOP，定义极大值为 MAX 以示区别）时，计数器会进行清零或减一操作。当计数器的计数值 TCNT1 到达最小值（等于 0x0000，定义为 BOTTOM）时，计数器会进行加一操作。当计数器的计数值 TCNT1 到达 OCR1A 或 OCR1B 时，也被称为发生比较匹配时，会清零或置位输出比较信号 OC1A 或 OC1B，来产生 PWM 波形。当使能插入死区时间时，设定的死区时间（DTR1 寄存器所对应的计数时钟数）将会插入到已产生的 PWM 波形中。当开启输入捕捉功能时，计数器被触发即开始或停止计数，ICR1 寄存器会记录捕捉信号触发周期内的计数值。软件可通过清除 COM1A/COM1B 位为零来关闭 OC1A/OC1B 的波形输出，或者设置相应的触发源，当触发事件发生时硬件自动清零 COM1A/COM1B 位来关闭 OC1A/OC1B 的波形输出。

计数时钟可由内部或外部时钟源来产生，时钟源的选择及分频选择由位于 TCCR1B 寄存器的 CS1 位来控制，详细描述见 TC0 和 TC1 预分频器章节。

计数器的长度为 16 位，支持双向计数。波形产生模式即计数器的工作模式由位于 TCCR1A 和 TCCR1B 寄存器的 WGM1 位来控制。根据不同的工作模式，计数器对每一个计数时钟 Clkt1 实现清零、加一或减一操作。当计数发生溢出时，位于 TIFR1 寄存器的计数溢出标志 TOV1 位会被置位。当中断使能时可产生 TC1 计数溢出中断。

输出比较单元对计数值 TCNT1 和输出比较寄存器 OCR1A 和 OCR1B 的值进行比较，当 TCNT1 等于 OCR1A 或 OCR1B 时称为发生比较匹配，位于 TIFR1 寄存器的输出比较标志 OCF1A 或 OCF1B 位会被置位。当中断使能时可产生 TC1 输出比较匹配中断。
需注意的是，在 PWM 工作模式下，OCR1A 和 OCR1B 寄存器为双缓冲寄存器。在普通模式和 CTC 模式下，双缓冲功能失效。计数到达最大值或最小值时，缓冲寄存器中的值被同步更新到比较寄存器 OCR1A 和 OCR1B 中去。详见工作模式章节描述。

波形产生器根据波形产生模式控制和比较输出模式控制使用比较匹配和计数溢出等来产生输出比较波形信号 OC1A 和 OC1B。具体产生方式见工作模式和寄存器章节描述。要把输出比较波形信号 OC1A 和 OC1B 输出到相应引脚上时，还必须设置该引脚的数据方向寄存器为输出。

工作模式

定时计数器 1 有六种不同的工作模式，包括普通模式（Normal），比较匹配时清零（CTC）模式，快速脉冲宽度调制（FPWM）模式，相位修正脉冲宽度调制（PCPWM）模式，相位频率修正脉冲宽度调制（PFCPWM）模式，和输入捕捉（ICP）模式。由波形产生模式控制位 WGM1[3:0]来选择。下面具体来描述这六种模式。由于有两个独立的输出比较单元，分别用“A”和“B”来表示，用小写的“x”来表示这两个输出比较单元通道。

普通模式

普通模式是定时计数器最简单的工作模式，此时波形产生模式控制位 WGM1[3:0]=0，计数的最大值 TOP 为 MAX（0xFFFF）。在这种模式下，计数方式为每一个计数时钟加一递增，当计数器到达 TOP 溢出后就回到 BOTTOM 重新开始累加。在计数值 TCNT1 变成零的同一个计数时钟里置位定时计数器溢出标志 TOV1。这种模式下 TOV1 标志就像是第 17 计数位，只是只会被置位不会被清零。溢出中断服务程序会自动清除 TOV1 标志，软件可以用它来提高定时计数器的分辨率。普通模式下没有特殊情形需要考虑，可以随时写入新的计数值。

设置 OC1x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC1x 的波形。当 COM1x=1 时，发生比较匹配时会翻转 OC1x 信号，这种情况下波形的频率可以用下面的公式来计算：

$f_{oc1xnormal} = f_{sys}/(2*N*65536)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

输出比较单元可以用来产生中断，但是在普通模式下不推荐使用中断，这样会占用太多 CPU 的时间。

CTC 模式

设置 WGM1[3:0]=4 或 12 时，定时计数器 1 进入 CTC 模式。当 WGM1[3]=0 时，计数最大值 TOP 为 OCR1A，当 WGM1[3]=1 时，计数最大值 TOP 为 ICR1。下面以 WGM1[3:0]=4 为例来描述 CTC 模式在这个模式下，计数方式为每一个计数时钟加一递增，当计数器的数值 TCNT1 等于 TOP 时计数器清零。这个模式使得用户可以很容易的控制比较匹配输出的频率，也简化了外部事件计数的操作。

当计数器到达 TOP 时，输出比较匹配标志 OCF1 被置位，相应的中断使能置位时将会产生中断。在中断服务程序里可以更新 OCR1A 寄存器。在这个模式下 OCR1A 没有使用双缓冲，在计数器以无预分频器或很低的预分频器工作下将最大值更新为接近最小值的时候要小心。如果写入 OCR1A 的数值小于当时的 TCNT1 值时，计数器将丢失一次比较匹配。在下一次比较匹配发生之前，计数器不得不先计数到 MAX，然后再从 BOTTOM 开始计数到 OCR1A。和普通模式一样，计数值回到 0x0 的计数时钟里置位 TOV1 标志。

设置 OC1x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC1x 的波形。波形的频率可以用下面的公式来计算：

$f_{oc1xctc} = f_{sys}/(2*N*(1+OCR1A))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

从公式可以看出，当设置 OCR1A 为 0x0 且无预分频器时，可以获得最大频率为 $f_{sys}/2$ 的输出波形。

当 WGM1[3:0]=12 时与 WGM1[3:0]=4 类似，只是把与 OCR1A 相关的换成 ICR1 即可。

快速 PWM 模式

设置 WGM1[3:0]=5，6，7，14 或 15 时，定时计数器 1 进入快速 PWM 模式，计数最大值 TOP 分别为 0xFF，0x1FF，0x3FF，ICR1 或 OCR1A，可以用来产生高频的 PWM 波形。快速 PWM 模式和其他 PWM 模式不同在于它是单向操作。计数器从 BOTTOM 累加到 TOP 后又回到 BOTTOM 重新计数。当计数值 TCNT1 到达 TOP 或 BOTTOM 时，输出比较信号 OC1x 会被置位或清零，取决于比较输出模式 COM1 的设置，详情见寄存器描述。由于采用单向操作，快速 PWM 模式的操作频率是采用双向操作的相位修正 PWM 模式的两倍。高频特性使得快速 PWM 模式适用于功率调节，整流以及 DAC 应用。高频信号可以减小外部元器件（电感电容等）的尺寸，从而降低系统成本。

当计数值到达 TOP 时，定时计数器溢出标志 TOV1 将会被置位，并把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新 OCR1A 寄存器。

设置 OC1x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC1x 的波形。波形的频率可用下面的公式来计算：

$f_{oc1xfpwm} = f_{sys}/(N*(1+TOP))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

当 TCNT1 和 OCR1x 发生比较匹配时，波形产生器就置位（清零）OC1x 信号，当 TCNT1 被清零时，波形产生器就清零（置位）OC1x 信号，以此来产生 PWM 波。由此 OCR1x 的极值将会产生特殊的 PWM 波形。当 OCR1x 设置为 0x00 时，输出的 PWM 为每(1+TOP)个计数时钟里有一个窄的尖峰脉冲。当 OCR1x 设置为 TOP 时，输出的波形为持续的高电平或低电平。如果用 OCR1A 作为 TOP 并设置 COM1A=1，输出比较信号 OC1A 会产生占空比为 50%的 PWM 波。

相位修正 PWM 模式

当设置 WGM0[3:0]=1，2，3，10 或 11 时，定时计数器 1 进入相位修正 PWM 模式，计数的最大值 TOP 分别为 0xFF, 0x1FF, 0x3FF, ICR1 或 OCR1A。计数器采用双向操作，由 BOTTOM 递增到 TOP，然后又递减到 BOTTOM，再重复此操作。计数到达 TOP 和 BOTTOM 时均改变计数方向，计数值在 TOP 或 BOTTOM 上均只停留一个计数时钟。在递增或递减过程中，计数值 TCNT1 与 OCR1x 匹配时，输出比较信号 OC1x 将会被清零或置位，取决于比较输出模式 COM1 的设置。与单向操作相比，双向操作可获得的最大频率要小，但其极好的对称性更适合于电机控制。

相位修正 PWM 模式下，当计数到达 BOTTOM 时置位 TOV1 标志，当计数到达 TOP 时把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR1x 存器。

设置 OC1x 脚的数据方向寄存器为输出时才能得到输出比较信号 OC1x 波形。波形的频率可用下面的公式来计算：

$f_{oc1xcpcpwm} = f_{sys}/(N*TOP*2)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

在递增计数过程中，当 TCNT1 与 OCR1x 匹配时，波形产生器就清零（置位）OC1x 信号。在

递减计数过程中，当 TCNT1 与 OCR1x 匹配时，波形产生器就置位（清零）OC1x 信号。由此 OCR1x 的极值会产生特殊的 PWM 波。当 OCR1x 设置为 TOP 或 BOTTOM 时，OC1x 信号输出会一直保持低电平或高电平。如果用 OCR1A 作为 TOP 并设置 COM1A=1，输出比较信号 OC1A 会产生占空比为 50%的 PWM 波。

为了保证输出 PWM 波在 BOTTOM 两侧的对称性，在没有发生比较匹配时，有两种情况下也会翻转 OC1x 信号。第一种情况是，当 OCR1x 的值由 TOP 改变为其他数据时。当 OCR1x 为 TOP，计数值达到 TOP 时，OC1x 的输出与前面降序计数时比较匹配的结果相同，即保持 OC1x 不变。此时会更新比较值为新的 OCR1x 的值（非 TOP），OC1x 的值会一直保持，直到升序计数时发生比较匹配而翻转。此时 OC1x 信号并不以最小值为中心对称，因此需要在 TCNT1 到达最大值时翻转 OC1x 信号，此即没有发生比较匹配时翻转 OC1x 信号的第一种情况。第二种情况是，当 TCNT1 从比 OCR1x 高的值开始计数时，因而会丢失一次比较匹配，从而引起不对称情形的产生。同样需要翻转 OC1x 信号去实现最小值两侧的对称性。

相位频率修正 PWM 模式

当设置 WGM0[3:0]=8 或 9 时，定时计数器 1 进入相位频率修正 PWM 模式，计数的最大值 TOP 分别为 ICR1 或 OCR1A。计数器采用双向操作，由 BOTTOM 递增到 TOP，然后又递减到 BOTTOM，再重复此操作。计数到达 TOP 和 BOTTOM 时均改变计数方向，计数值在 TOP 或 BOTTOM 上均只停留一个计数时钟。在递增或递减过程中，计数值 TCNT1 与 OCR1x 匹配时，输出比较信号 OC1x 将会被清零或置位，取决于比较输出模式 COM1 的设置。与单向操作相比，双向操作可获得的最大频率要小，但其极好的对称性更适合于电机控制。

相位频率修正 PWM 模式下，当计数到达 BOTTOM 时置位 TOV1 标志，并且把比较缓冲器的值更新到比较值，更新比较值的时间是相位频率修正 PWM 模式和相位修正 PWM 模式的最大不同点。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR1x 存器。当 CPU 改变 TOP 值即 ORC1A 或 ICR1 的值时，必须保证新的 TOP 值不小于已经在使用的 TOP 值，否则比较匹配将不会再发生。

设置 OC1x 脚的数据方向寄存器为输出时才能得到输出比较信号 OC1x 波形。波形的频率可用下面的公式来计算：

$f_{oc1xcpfcpwm} = f_{sys}/(N*TOP*2)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

在递增计数过程中，当 TCNT1 与 OCR1x 匹配时，波形产生器就清零（置位）OC1x 信号。在递减计数过程中，当 TCNT1 与 OCR1x 匹配时，波形产生器就置位（清零）OC1x 信号。由此 OCR1x 的极值会产生特殊的 PWM 波。当 OCR1x 设置为 TOP 或 BOTTOM 时，OC1x 信号输出会一直保持低电平或高电平。如果用 OCR1A 作为 TOP 并设置 COM1A=1，输出比较信号 OC1A 会产生占空比为 50%的 PWM 波。

因为 OCR1x 寄存器是在 BOTTOM 时刻更新的，所以 TOP 值两边升序和降序的计数长度是一样的，也就产生了频率和相位都正确的对称波形。

当使用固定 TOP 值时，最好采用 ICR1 寄存器作为 TOP 值，即设置 WGM1[3:0]=8，此时 OCR1A 寄存器只需用来产生 PWM 输出。如果要产生频率变化的 PWM 波，必须通过改变 TOP 值，

OCR1A 的双缓冲特性会更适合于这个应用。

输入捕捉模式

输入捕捉用来捕获外部事件，并为其赋予时间标记以说明此事件发生的时刻，可以在前面的计数模式下进行，不过要除去使用 ICR1 值作为计数 TOP 值的波形产生模式。

外部事件发生的触发信号由引脚 ICP1 输入，也可以通过模拟比较器单元来实现。当引脚 ICP1 上的逻辑电平发生变化，或模拟比较器的输出 ACO 电平发生变化，并且这个电平变化被输入捕捉单元所捕获，输入捕捉即被触发，此时 16 位的计数值 TCNT1 数据被复制到输入捕捉寄存器 ICR1，同时输入捕捉标志 ICF1 置位，若 ICIE1 位为“1”，输入捕捉标志将产生输入捕捉中断。

通过设置模拟比较控制与状态寄存器 ACSR 的模拟比较输入捕捉控制位 ACIC 来选择输入捕捉触发源 ICP1 或 ACO。需注意的是，改变触发源有可能造成一次输入捕捉，因此在改变触发源后必须对 ICF1 进行一次清零操作来避免出现错误的结果。

输入捕捉信号经过一个可选的噪声抑制器之后送入边沿检测器，根据输入捕捉选择控制位 ICES1 的配置，看检测到的边沿是否满足触发条件。噪声抑制器是一个简单的数字滤波，对输入信号进行 4 次采样，只有当 4 次采样值都相等时其输出才会送入边沿检测器。噪声抑制器由 TCCR1B 寄存器的 ICNC1 位控制其使能或禁止。

使用输入捕捉功能时，当 ICF1 被置位后，应尽可能早的读取 ICR1 寄存器的值，因为下一次捕捉事件发生后 ICR1 的值将会被更新。推荐使能输入捕捉中断，在任何输入捕捉工作模式下，都不推荐在操作过程中改变计数 TOP 值。

输入捕捉到的时间标记可用来计算频率、占空比及信号的其它特征，以及为触发事件创建日志。测量外部信号的占空比时要求每次捕捉后都要改变触发沿，因此读取 ICR1 值以后须尽快改变触发的信号边沿。

PWM 输出的自动关闭与重启

当设置 TCCR1C 寄存器的 DOC1x 位为高时，PWM 输出的自动关闭功能会被使能，满足触发条件时，硬件会清零相应的 COM1x 位，将 PWM 输出信号 OC1x 与其输出引脚断开，切换成通用 IO 输出，实现 PWM 输出的自动关闭。此时，输出引脚的状态可由通用 IO 口的输出来控制。

PWM 输出的自动关闭被使能后，还需要设置其触发条件，由 TCCR1D 寄存器的 DSX1n 位来选择触发源。触发源有模拟比较器中断，外部中断，引脚电平变化中断以及定时器溢出中断，具体情形请参考 TCCR1D 寄存器描述。当某个或某些触发源被选用作为触发条件后，在这些中断标志位被置位的同时，硬件会清零 COM1x 位来关闭 PWM 的输出。

当发生了触发事件关闭 PWM 输出后，定时器模块没有相应的中断标志位，软件需要通过读取触发源的中断标志位来得知触发条件和触发事件。

当 PWM 输出被自动关闭而需要再次重启输出时，软件只需要重新设置 COM1x 位，来切换

OC1x 信号输出到相应的引脚上。需要注意的是，发生自动关闭后，定时器并未停止工作，OC1x 信号的状态也一直在更新。软件可在定时器发生溢出或比较匹配后，再设置 COM1x 位来输出 OC1x 信号，这样可以获得明确的 PWM 输出状态。

死区时间控制

设置 DTEN1 位为“1”时，插入死区时间的功能被使能，OC1A 和 OC1B 的输出波形将在 B 通道比较输出所产生的波形基础上插入设定的死区时间，时间的长度为 DTR1 寄存器的计数时钟数所对应的时间值。如下图所示，OC1A 和 OC1B 的死区时间插入均是以通道 B 的比较输出波形为基准。当 COM1A 和 COM1B 同为“2”或“3”时，OC1A 的波形极性与 OC1B 的波形极性相同，当 COM1A 和 COM1B 分别为“2”或“3”时，OC1A 的波形与 OC1B 的波形极性相反。

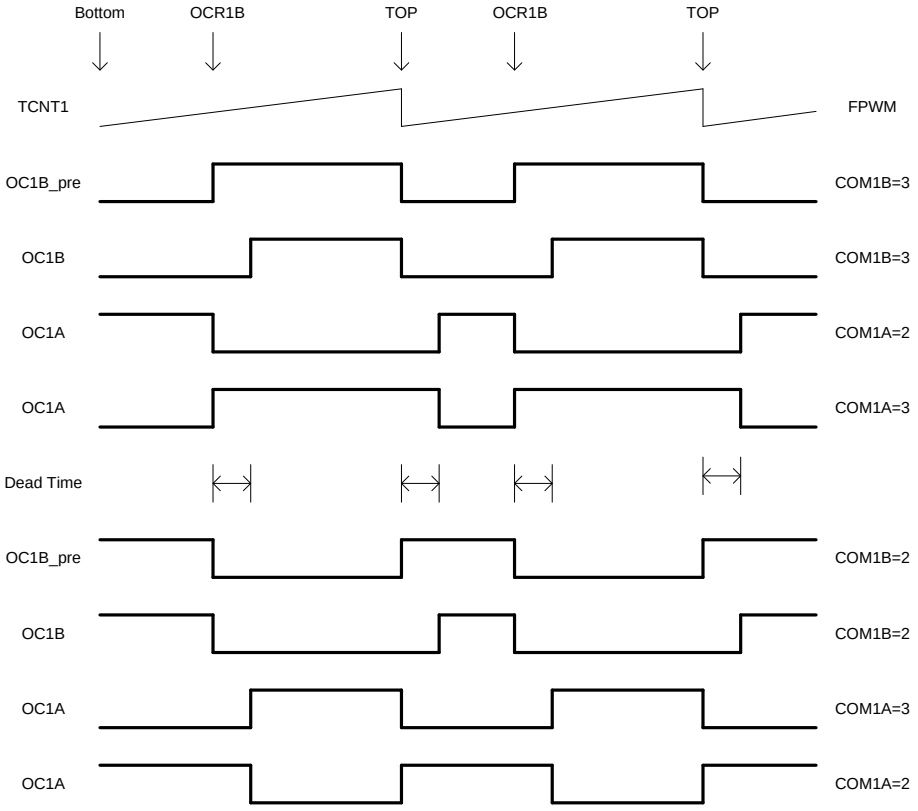


Figure 3 FPWM 模式下 TC1 死区时间控制

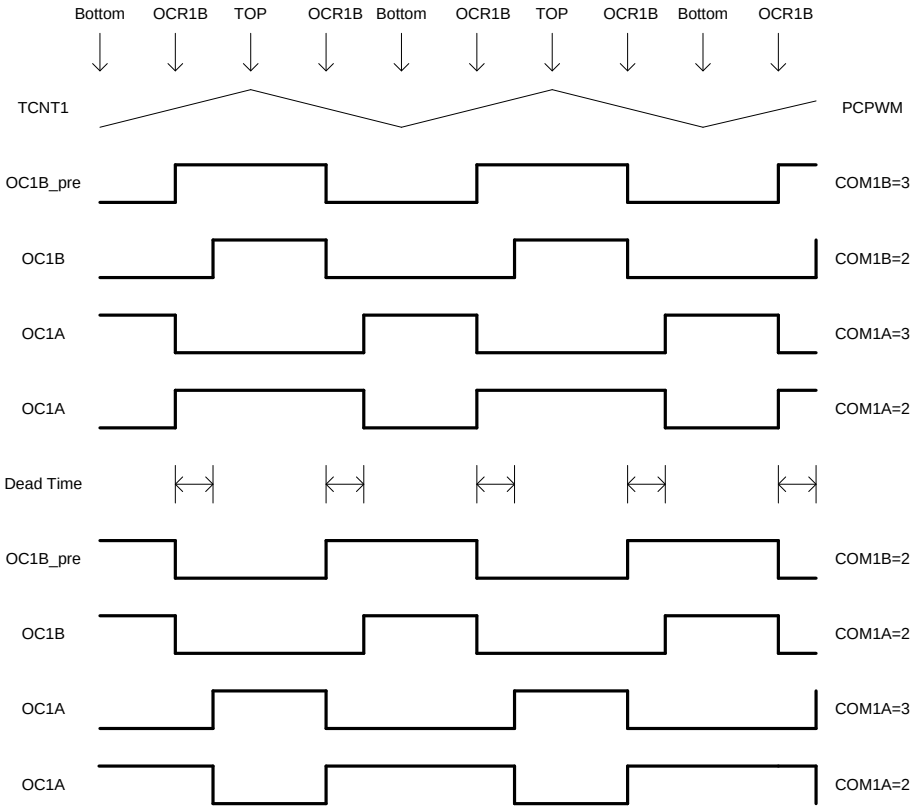


Figure 4 PCPWM 模式下 TC1 死区时间控制

设置 DTEN1 位为“0”时，插入死区时间的功能被禁止，OC1A 和 OC1B 的输出波形为各自比较输出所产生的波形。

高速计数模式

高速时钟模式下，采用更高频率的时钟作为计数的时钟源，用来产生更高速度和更高分辨率的 PWM 波形。此高频时钟是通过对内部 32M RC 振荡器的输出时钟 rc32m 进行 2 倍频来产生的。因此，在进入高频模式之前，需先使能内部 32M RC 振荡器的倍频功能，即置位 TCKCSR 寄存器的 F2XEN 位，并等待一定时间直到倍频时钟信号输出稳定。然后，可置位 TCKCSR 的 TC2XS1 位来使定时计数器进入高速时钟模式。

在此模式下，系统时钟与高速时钟是异步关系，而部分寄存器（见 TC1 寄存器列表）是工作在高速时钟域，因此，配置和读取这类寄存器时也是异步的，操作时需注意。

对高速时钟域下的寄存器进行非连续读写操作时无特殊要求，而进行连续读写操作时，需等待一个系统时钟，可按以下步骤：

5） 写寄存器 A；
6） 等待一个系统时钟（NOP 或操作系统时钟下的寄存器）；
7） 读或写寄存器 A 或 B。
8） 等待一个系统时钟（NOP 或操作系统时钟下的寄存器）。

对高速时钟域下的寄存器进行读操作时，宽度为 8 位的寄存器均可直接读取，而读取 16 位寄存器的值（OCR1A，OCR1B, ICR1, TCNT1）时，先读取低位寄存器的值，等待一个系统时钟

后，再读取高位寄存器的值，而在读取 TCNT1 的值时，当计数器还在进行计数时，TCNT1 的值会随高速时钟变化，可暂停计数器（设置 CS1 为零）再读取 TCNT1 的值。

读取 OCR1A，OCR1B 和 ICR1 时，可按以下步骤：
1） 读取 OCR1AL/OCR1BL/ICR1L；
2） 等待一个系统时钟（NOP）；
3） 读取 OCR1AH/OCR1BH/ICR1H。

读取 TCNT1 时，可按以下步骤：
1） 置 CS1 为零；
2） 等待一个系统时钟（NOP）；
3） 读取 TCNT1L 的值；
4） 等待一个系统时钟（NOP）；
读取 TCNT1H 的值。

寄存器定义

TC1 寄存器列表

寄存器	地址	默认值	描述
TCCR1A*	0x80	0x00	TC1 控制寄存器 A
TCCR1B*	0x81	0x00	TC1 控制寄存器 B
TCCR1C*	0x82	0x00	TC1 控制寄存器 C
DSX1	0x83	0x00	TC1 触发源控制寄存器
TCNT1L*	0x84	0x00	TC1 计数值寄存器低字节
TCNT1H*	0x85	0x00	TC1 计数值寄存器高字节
ICR1L*	0x86	0x00	TC1 输入捕捉寄存器低字节
ICR1H*	0x87	0x00	TC1 输入捕捉寄存器高字节
OCR1AL*	0x88	0x00	TC1 输出比较寄存器 A 低字节
OCR1AH*	0x89	0x00	TC1 输出比较寄存器 A 高字节
OCR1BL*	0x8A	0x00	TC1 输出比较寄存器 B 低字节
OCR1BH*	0x8B	0x00	TC1 输出比较寄存器 B 高字节
DTR1*	0x8C	0x00	TC1 死区时间控制寄存器
TIMSK1	0x6F	0x00	定时计数器中断屏蔽寄存器
TIFR1	0x36	0x00	定时计数器中断标志寄存器
TCKCSR1	0xEC	0x00	TC1 时钟控制状态寄存器

【注意】
带“*”的寄存器工作于系统时钟和高速时钟域下，未带“*”的寄存器仅工作于系统时钟域下。

TCCR1A –TC1 控制寄存器 A

TCCR1A –TC1 控制寄存器 A								
地址: 0x80					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	COM1A1	COM1A0	COM1B1	COM1B0	-	-	WGM11	WGM10
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	描述
7	COM1A1	比较匹配输出 A 模式控制高位。 COM1A1 和 COM1A0 组成 COM1A[1:0]来控制输出比较波形 OC1A。如果 COM1A 的 1 位或者 2 位都置位，输出比较波形占据着 OC1A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM1A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
6	COM1A0	比较匹配输出 A 模式控制低位。 COM1A1 和 COM1A0 组成 COM1A[1:0]来控制输出比较波形 OC1A。如果 COM1A 的 1 位或者 2 位都置位，输出比较波形占据着 OC1A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM1A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
5	COM1B1	比较匹配输出 B 模式控制高位。 COM1B1 和 COM1B0 组成 COM1B[1:0]来控制输出比较波形 OC1B。如果 COM1B 的 1 位或者 2 位都置位，输出比较波形占据着 OC1B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM1B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
4	COM1B0	比较匹配输出 B 模式控制低位。 COM1B1 和 COM1B0 组成 COM1B[1:0]来控制输出比较波形 OC1B。如果 COM1B 的 1 位或者 2 位都置位，输出比较波形占据着 OC1B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM1B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
3:2	-	保留不变
1	WGM11	波形产生模式控制次低位。 WGM11 和 WGM13,WGM12,WGM10 一起组成波形产生模式控制 WGM1[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
0	WGM10	波形产生模式控制最低位。 WGM10 和 WGM13,WGM12,WGM11 一起组成波形产生模式控制 WGM1[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。

下表为非 PWM 模式（即普通模式和 CTC 模式）下，比较输出模式对输出比较波形的控制。

COM1x[1:0]	描述
0	OC1x 断开，通用 IO 口操作
1	比较匹配时翻转 OC1x 信号
2	比较匹配时清零 OC1x 信号
3	比较匹配时置位 OC1x 信号

下表为快速 PWM 模式下比较输出模式对输出比较波形的控制。

COM1x[1:0]	描述
0	OC1x 断开，通用 IO 口操作
1	WGM1 为 15 时：比较匹配时翻转 OC1A 信号, OC1B 断开 WGM1 为其它值时：OC1x 断开，通用 IO 口操作
2	比较匹配时清零 OC1x 信号，最大值匹配时置位 OC1x 信号
3	比较匹配时置位 OC1x 信号，最大值匹配时清零 OC1x 信号

下表为相位修正模式下比较输出模式对输出比较波形的控制。

COM1x[1:0]	描述
0	OC1x 断开，通用 IO 口操作
1	WGM1 为 9 或 11 时：比较匹配时翻转 OC1A 信号, OC1B 断开 WGM1 为其它值时：OC1x 断开，通用 IO 口操作
2	升序计数下比较匹配清零 OC1x 信号，降序计数下比较匹配置位 OC1x 信号
3	升序计数下比较匹配置位 OC1x 信号，降序计数下比较匹配清零 OC1x 信号

TCCR1B –TC1 控制寄存器 B

TCCR1B–TC1 控制寄存器 B								
地址: 0x81					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	ICNC1	输入捕捉噪声抑制器使能控制位。 当设置 ICNC1 位为“1”时，使能输入捕捉噪声抑制器，此时外部引脚 ICP1 的输入被滤波，连续 4 个采样值相等时输入信号才有效，该功能使得输入捕捉被延迟了 4 个时钟周期。 当设置 ICNC1 位为“0”时，禁止输入捕捉噪声抑制器，此时外部引脚 ICP1 的输入直接有效。
6	ICES1	输入捕捉触发沿选择控制位。 当设置 ICES1 位为“1”时，选择电平的上升沿触发输入捕捉；当设置 ICES1 位为“0”时，选择电平的下降沿触发输入捕捉。 当捕获到一个事件后，计数器的数值被复制到 ICR1 寄存器，同时置位输入捕捉标志 ICF1。如果中断使能，产生输入捕捉中断。
5	-	保留。

<table>
<tr><td>4</td><td>WGM13</td><td>波形产生模式控制高位。
WGM13 和 WGM12,WGM11,WGM10 一起组成波形产生模式控制 WGM1[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。</td></tr>
<tr><td>3</td><td>WGM12</td><td>波形产生模式控制次高位。
WGM12 和 WGM13,WGM11,WGM10 一起组成波形产生模式控制 WGM1[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。</td></tr>
<tr><td>2</td><td>CS12</td><td>时钟选择控制高位。用于选择定时计数器 1 的时钟源。</td></tr>
<tr><td>1</td><td>CS11</td><td>时钟选择控制中位。用于选择定时计数器 1 的时钟源。</td></tr>
<tr><td>0</td><td>CS10</td><td>时钟选择控制低位。
用于选择定时计数器 1 的时钟源。
<table>
<tr><th>CS1[2:0]</th><th>描述</th></tr>
<tr><td>0</td><td>无时钟源，停止计数</td></tr>
<tr><td>1</td><td>clk_{sys}</td></tr>
<tr><td>2</td><td>clk_{sys}/8，来自预分频器</td></tr>
<tr><td>3</td><td>clk_{sys}/64，来自预分频器</td></tr>
<tr><td>4</td><td>clk_{sys}/256，来自预分频器</td></tr>
<tr><td>5</td><td>clk_{sys}/1024，来自预分频器</td></tr>
<tr><td>6</td><td>外部时钟 T1 引脚，下降沿触发</td></tr>
<tr><td>7</td><td>外部时钟 T1 引脚，上升沿触发</td></tr>
</table></td></tr>
</table>

下表为波形产生模式控制。

WGM1[3:0]	工作模式	TOP 值	更新 OCR0 时刻	置位 TOV0 时刻
0	Normal	0xFFFF	立即	MAX
1	8 位 PCPWM	0x00FF	TOP	BOTTOM
2	9 位 PCPWM	0x01FF	TOP	BOTTOM
3	10 位 PCPWM	0x03FF	TOP	BOTTOM
4	CTC	OCR1A	立即	MAX
5	8 位 FPWM	0x00FF	BOTTOM	TOP
6	9 位 FPWM	0x01FF	BOTTOM	TOP
7	10 位 FPWM	0x03FF	BOTTOM	TOP
8	PFCPWM	ICR1	BOTTOM	BOTTOM
9	PFCPWM	OCR1A	BOTTOM	BOTTOM
10	PCPWM	ICR1	TOP	BOTTOM
11	PCPWM	OCR1A	TOP	BOTTOM
12	CTC	ICR1	立即	MAX
13	保留	-	-	-
14	FPWM	ICR1	TOP	TOP
15	FPWM	OCR1A	TOP	TOP

TCCR1C –TC1 控制寄存器 C

TCCR1C–TC1 控制寄存器 C								
地址: 0x82					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	FOC1A	FOC1B	DOC1B	DOC1A	DTEN1	-	-	-
R/W	W	W	R/W	R/W	R/W	-	-	-

Bit	Name	描述
7	FOC1A	强制输出比较 A。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC1A 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF1A 标志，也不会重载或清零定时器，但是输出引脚 OC1A 将被按照 COM1A 的设置相应的更新，就跟真的发生了比较匹配一样。 工作于 PWM 模式时，写 TCCR1A 寄存器时要对其清零。 读取 FOC1A 的返回值一直为零。
6	FOC1B	强制输出比较 B。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC1B 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF1B 标志，也不会重载或清零定时器，但是输出引脚 OC1B 将被按照 COM1B 的设置相应的更新，就跟真的发生了比较匹配一样。工作于 PWM 模式时，写 TCCR1A 寄存器时要对其清零。读取 FOC1B 的返回值一直为零。
5	DOC1B	TC1 关闭输出比较使能控制高位。 当设置 DOC1B 位为“1”时，触发源关闭输出比较信号 OC1B 被使能。当发生触发事件时，硬件自动清零 COM1B 位，关闭 OC1B 的波形输出。软件可通过设置 COM1B 重新开启 PWM 输出。 当设置 DOC1B 位为“0”时，触发源关闭输出比较信号 OC1B 被禁止。
4	DOC1A	TC1 关闭输出比较使能控制低位。 当设置 DOC1A 位为“1”时，触发源关闭输出比较信号 OC1A 被使能。当发生触发事件时，硬件自动清零 COM1A 位，关闭 OC1A 的波形输出。软件可通过设置 COM1A 重新开启 PWM 输出。 当设置 DOC1A 位为“0”时，触发源关闭输出比较信号 OC1A 被禁止。
3	DTEN1	TC1 死区时间使能控制位。 当设置 DTEN1 位为“1”时，使能死区时间插入。OC1A 和 OC1B 均在 B 通道比较输出产生的波形基础上插入死区时间，所插入的死区时间间隔由 DTR1 寄存器所对应的计数时间决定。OC1A 输出波形的极性由 COM1A 和 COM1B 的对应关系决定，详见 OC1A 插入死区时间后波形极性表格所示。 当设置 DTEN1 位为“0”时，禁止死区时间插入，OC1A 和 OC1B 的波形为各自比较输出所产生的波形。
2:0	-	保留

下表为死区时间使能时 OC1A 信号输出波形的极性控制。

死区时间使能模式下 OC1A 信号输出波形的极性控制

DTEN1	COM1A[1:0]	COM1B[1:0]	描述
0	-	-	OC1A 信号极性由 OC1A 比较输出模式控制
1	0	-	OC1A 断开，通用 IO 口操作
1	1	-	保留
1	2	2	OC1A 信号与 OC1B 信号极性相同
		3	OC1A 信号与 OC1B 信号极性相反
1	3	2	OC1A 信号与 OC1B 信号极性相反
		3	OC1A 信号与 OC1B 信号极性相同

【注意】:
OC1B 信号输出波形的极性由 OC1B 比较输出模式控制，与未使能死区时间模式相同。

TCCR1D –TC1 控制寄存器 D

TCCR1D –TC 控制寄存器 D								
地址: 0x83					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	DSX17	DSX16	DSX15	DSX14	-	-	DSX11	DSX10
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	描述
7	DSX17	TC1 触发源选择控制使能第 7 位。 当设置 DSX17 位为“1”时，TC0 溢出作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被使能。当 DOC1A/DOC1B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC1A/OC1B 的波形输出。 当设置 DSX17 位为“0”时，TC0 溢出作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被禁止。
6	DSX16	TC1 触发源选择控制使能第 6 位。 当设置 DSX16 位为“1”时，TC2 溢出作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被使能。当 DOC1A/DOC1B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC1A/OC1B 的波形输出。 当设置 DSX16 位为“0”时，TC2 溢出作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被禁止。
5	DSX15	TC1 触发源选择控制使能第 5 位。 当设置 DSX15 位为“1”时，引脚电平变化 1 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被使能。当 DOC1A/DOC1B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC1A/OC1B 的波形输出。 当设置 DSX15 位为“0”时，引脚电平变化 1 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被禁止。
4	DSX14	TC1 触发源选择控制使能第 4 位。 当设置 DSX14 位为“1”时，外部中断 1 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被使能。当 DOC1A/DOC1B 位为

		“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC1A/OC1B 的波形输出。 当设置 DSX14 位为“0”时，外部中断 1 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被禁止。
3:2	-	保留
1	DSX11	TC1 触发源选择控制使能第 1 位。 当设置 DSX11 位为“1”时，模拟比较器 1 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被使能。当 DOC1A/DOC1B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC1A/OC1B 的波形输出。 当设置 DSX11 位为“0”时，模拟比较器 1 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被禁止。
0	DSX10	TC1 触发源选择控制使能第 0 位。 当设置 DSX10 位为“1”时，模拟比较器 0 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被使能。当 DOC1A/DOC1B 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC1A/OC1B 的波形输出。 当设置 DSX10 位为“0”时，模拟比较器 0 作为为关闭输出比较信号波形 OC1A/OC1B 的触发源被禁止。

下表为波形输出的触发源的选择控制。

关闭 OC1A/OC1B 波形输出的触发源选择控制

DOC1x	DSX1n=1	触发源	描述
0	-	-	DOC1x 位为“0”，触发源关闭波形输出功能被禁止
1	0	模拟比较器 0	ACIF0 的上升沿将关闭 OC1x 波形输出
1	1	模拟比较器 1	ACIF1 的上升沿将关闭 OC1x 波形输出
1	4	外部中断 1	INTF1 的上升沿将关闭 OC1x 波形输出
1	5	引脚电平变化 1	PCIF1 的上升沿将关闭 OC1x 波形输出
1	6	TC2 溢出	TOV2 的上升沿将关闭 OC1x 波形输出
1	7	TC0 溢出	TOV0 的上升沿将关闭 OC1x 波形输出

【注意】:

DSX1n=1 表示 DSX1 寄存器的第 n 位为 1 时，各寄存器位可同时置位。

TCNT1L –TC1 计数值寄存器低字节

TCNT1L –TC1 计数值寄存器低字节								
地址: 0x84					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT1L7	TCNT1L6	TCNT1L5	TCNT1L4	TCNT1L3	TCNT1L2	TCNT1L1	TCNT1L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	TCNT1	TC1 计数值的低字节。						

		TCNT1H 和 TCNT1L 结合到一起组成 TCNT1，通过 TCNT1 寄存器可以直接对计数器的 16 位计数值进行读写访问。读写 16 位寄存器需要两次操作。写 16 位 TCNT1 时，应先写入 TCNT1H。读 16 位 TCNT1 时，应先读取 TCNT1L。 CPU 对 TCNT1 寄存器的写操作会在下一个定时器时钟周期阻止比较匹配的发生，即使定时器已经停止。这就允许初始化 TCNT1 寄存器的值与 OCR1x 的值一致而不会引发中断。 如果写入 TCNT1 的数值等于或绕过 OCR1x 值时，比较匹配就会丢失，造成不正确的波形发生结果。 没有选择时钟源时定时器停止计数，但 CPU 仍可以访问 TCNT1。 CPU 写计数器比清零或加减操作的优先级高。

TCNT1H –TC1 计数值寄存器高字节

TCNT1H –TC1 计数值寄存器高字节								
地址: 0x85					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT1H7	TCNT1H6	TCNT1H5	TCNT1H4	TCNT1H3	TCNT1H2	TCNT1H1	TCNT1H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	TCNT1H	TC1 计数值的高字节。 TCNT1H 和 TCNT1L 结合到一起组成 TCNT1，通过 TCNT1 寄存器可以直接对计数器的 16 位计数值进行读写访问。读写 16 位寄存器需要两次操作。写 16 位 TCNT1 时，应先写入 TCNT1H。读 16 位 TCNT1 时，应先读取 TCNT1L。 CPU 对 TCNT1 寄存器的写操作会在下一个定时器时钟周期阻止比较匹配的发生，即使定时器已经停止。这就允许初始化 TCNT1 寄存器的值与 OCR1x 的值一致而不会引发中断。 如果写入 TCNT1 的数值等于或绕过 OCR1x 值时，比较匹配就会丢失，造成不正确的波形发生结果。 没有选择时钟源时定时器停止计数，但 CPU 仍可以访问 TCNT1。 CPU 写计数器比清零或加减操作的优先级高。							

ICR1L –TC1 输入捕捉寄存器低字节

ICR1L –TC1 输入捕捉寄存器低字节								
地址: 0x86					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICR1L7	ICR1L6	ICR1L5	ICR1L4	ICR1L3	ICR1L2	ICR1L1	ICR1L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	ICR1L	TC1 输入捕捉值的低字节。 ICR1H 和 ICR1L 结合到一起组成 16 位的 ICR1。读写 16 位寄存器需要两次操作。写 16 位 ICR1 时，应先写入 ICR1H。读 16 位							

		ICR1 时，应先读取 ICR1L。当输入捕捉被触发时，计数值 TCNT1 就会更新复制到 ICR1 寄存器里。ICR1 寄存器也可用来定义计数的 TOP 值。

ICR1H –TC1 输入捕捉寄存器高字节

ICR1H –TC1 输入捕捉寄存器高字节								
地址: 0x87					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICR1H7	ICR1H6	ICR1H5	ICR1H4	ICR1H3	ICR1H2	ICR1H1	ICR1H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	ICR1H	TC1 输入捕捉值的高字节。 ICR1H 和 ICR1L 结合到一起组成 16 位的 ICR1。读写 16 位寄存器需要两次操作。写 16 位 ICR1 时，应先写入 ICR1H。读 16 位 ICR1 时，应先读取 ICR1L。当输入捕捉被触发时，计数值 TCNT1 就会更新复制到 ICR1 寄存器里。ICR1 寄存器也可用来定义计数的 TOP 值。						

OCR1AL –TC1 输出比较寄存器 A 低字节

OCR1AL –TC1 输出比较寄存器 A 低字节								
地址: 0x88					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1AL7	OCR1AL6	OCR1AL5	OCR1AL4	OCR1AL3	OCR1AL2	OCR1AL1	OCR1AL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR1AL	输出比较寄存器 A 的低字节。 OCR1AL 和 OCR1AH 结合到一起组成 16 位的 OCR1A。读写 16 位寄存器需要两次操作。写 16 位 OCR1A 时，应先写入 OCR1AH。读 16 位 OCR1A 时，应先读取 OCR1AL。 OCR1A 不间断地与计数器数值 TCNT1 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC1A 引脚上产生波形。 当使用 PWM 模式时，OCR1A 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR1A 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR1A 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR1A 本身。							

OCR1AH –TC1 输出比较寄存器 A 高字节

OCR1AH –TC1 输出比较寄存器 A 高字节								
地址: 0x89					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1AH7	OCR1AH6	OCR1AH5	OCR1AH4	OCR1AH3	OCR1AH2	OCR1AH1	OCR1AH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR1AH	输出比较寄存器 A 的高字节。 OCR1AL 和 OCR1AH 结合到一起组成 16 位的 OCR1A。读写 16 位寄存器需要两次操作。写 16 位 OCR1A 时，应先写入 OCR1AH。读 16 位 OCR1A 时，应先读取 OCR1AL。 OCR1A 不间断地与计数器数值 TCNT1 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC1A 引脚上产生波形。 当使用 PWM 模式时，OCR1A 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR1A 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR1A 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR1A 本身。							

OCR1BL –TC1 输出比较寄存器 B 低字节

OCR1BL –TC1 输出比较寄存器 B 低字节								
地址: 0x8A					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1BL7	OCR1BL6	OCR1BL5	OCR1BL4	OCR1BL3	OCR1BL2	OCR1BL1	OCR1BL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR1BL	输出比较寄存器 B 的低字节。 OCR1BL 和 OCR1BH 结合到一起组成 16 位的 OCR1B。读写 16 位寄存器需要两次操作。写 16 位 OCR1B 时，应先写入 OCR1BH。读 16 位 OCR1B 时，应先读取 OCR1BL。 OCR1B 不间断地与计数器数值 TCNT1 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC1B 引脚上产生波形。当使用 PWM 模式时，OCR1B 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR1B 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR1B 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR1B 本身。							

OCR1BH –TC1 输出比较寄存器 B 高字节

OCR1BH –TC1 输出比较寄存器 B 高字节								
地址: 0x8B					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR1BH7	OCR1BH6	OCR1BH5	OCR1BH4	OCR1BH3	OCR1BH2	OCR1BH1	OCR1BH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR1BH	输出比较寄存器 B 的高字节。 OCR1BL 和 OCR1BH 结合到一起组成 16 位的 OCR1B。读写 16 位寄存器需要两次操作。写 16 位 OCR1B 时，应先写入 OCR1BH。读 16 位 OCR1B 时，应先读取 OCR1BL。 OCR1B 不间断地与计数器数值 TCNT1 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC1B 引脚上产生波形。 当使用 PWM 模式时，OCR1B 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR1B 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR1B 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR1B 本身。							

TIMSK1 – TC1 中断屏蔽寄存器

TIMSK1 – TC1 中断屏蔽寄存器								
地址: 0x6F					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	TICIE1	-	-	OCIE1A	OCIE1B	TOIE1
R/W	-	-	R/W	-	-	R/W	R/W	R/W
Bit	Name	描述						
7:6	-	保留。						
5	TICIE1	TC1 输入捕捉中断使能控制位。 当 ICIE1 位为“1”时，且全局中断置位，TC1 输入捕捉中断被使能。当输入捕捉触发时，即 TIFR1 的 ICF1 标志被置位，中断发生。 当 ICIE1 位为“0”时，TC1 输入捕捉中断被禁止。						
4:3	-	保留。						
2	OCIE1B	TC1 输出比较 B 匹配中断使能位。 当 OCIE1B 位为“1”，且全局中断置位，TC1 输出比较 B 匹配中断使能。当比较匹配发生时，即 TIFR 中 OCF1B 位被置位时，中断产生。 当 OCIE1B 位为“0”时，TC1 输出比较 B 匹配中断被禁止。						
1	OCIE1A	TC1 输出比较 A 匹配中断使能位。 当 OCIE1A 位为“1”，且全局中断置位，TC1 输出比较 A 匹配中						

		断使能。当比较匹配发生时，即 TIFR 中 OCF1A 位被置位时，中断产生。 当 OCIE1A 位为“0”时，TC1 输出比较 A 匹配中断被禁止。
0	TOIE1	TC1 溢出中断使能位。 当 TOIE1 位为“1”，且全局中断置位，TC1 溢出中断使能。当 TC1 发生溢出，即 TIFR 中的 TOV1 位被置位时，中断产生。 当 TOIE1 位为“0”时，TC1 溢出中断被禁止。

TIFR1 – TC1 中断标志寄存器

TIFR1 – TC1 中断标志寄存器								
地址: 0x36					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	ICF1	-	-	OCF1B	OCF1A	TOV1
R/W	-	-	R/W	-	-	R/W	R/W	R/W

Bit	Name	描述
7:6	-	保留。
5	ICF1	输入捕捉标志位。 当输入捕捉事件发生时，ICF1 标志被置位。当 ICR1 被用作计数的 TOP 值，且计数值到达 TOP 值时，ICF1 标志被置位。若 ICIE1 为“1”且全局中断标志置位，则会产生输入捕捉中断。执行此中断服务程序时 ICF1 将自动清零，或对 ICF1 位写“1”也可清零该位。
4:3	-	保留。
2	OCF1B	输出比较 B 匹配标志位。 当 TCNT1 等于 OCR1B 时，比较单元就给出匹配信号，并置位比较标志 OCF1B。若此时输出比较中断使能 OCIE1B 为“1”且全局中断标志置位，则会产生输出比较中断。执行此中断服务程序时 OCF1B 将自动清零，或对 OCF1B 位写“1”也可清零该位。
1	OCF1A	输出比较 A 匹配标志位。 当 TCNT1 等于 OCR1A 时，比较单元就给出匹配信号，并置位比较标志 OCF1A。若此时输出比较中断使能 OCIE1A 为“1”且全局中断标志置位，则会产生输出比较中断。执行此中断服务程序时 OCF1A 将自动清零，或对 OCF1A 位写“1”也可清零该位。
0	TOV1	溢出标志位。 当计数器发生溢出时，置位溢出标志 TOV1。若此时溢出中断使能 TOIE1 为“1”且全局中断标志置位，则会产生溢出中断。执行此中断服务程序时 TOV1 将自动清零，或对 TOV1 位写“1”也可清零该位。

DTR1L –TC1 死区时间寄存器低字节

DTR1–TC1 死区时间寄存器								
地址: 0x8C					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0

	DTR1L							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	DTR1L	死区时间寄存器高字节。 当 DTEN1 位为高时，OC1A 和 OC1B 为互补输出，OC1A 输出上所插入的死区时间由 DTR1L 个计数时钟决定。						

DTR1H –TC1 死区时间寄存器高字节

DTR1H–TC1 死区时间寄存器高字节								
地址: 0x8D					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	DTR1H							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	DTR1H	死区时间寄存器高字节。 当 DTEN1 位为高时，OC1A 和 OC1B 为互补输出，OC1B 输出上所插入的死区时间由 DTR1H 个计数时钟决定。						

TCKCSR –TC 时钟控制状态寄存器

TCKCSR–TC 时钟控制状态寄存器								
地址: 0xEC					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	F2XEN	TC2XF1	TC2XF0	-	AFCKS	TC2XS1	TC2XS0
R/W	-	R/W	R/0	R/0	-	R/W	R/W	R/W
Bit	Name	描述						
7	-	保留						
6	F2XEN	RC 32M 倍频输出使能控制位 当设置 F2XEN 位为“1”时，32M RC 振荡器的倍频输出被使能，输出 64M 的高速时钟 当设置 F2XEN 位为“1”时，32M RC 振荡器的倍频输出被禁止，不能输出 64M 的高速时钟						
5	TC2XF1	TC 高速时钟模式标志位 1 当读到 TC2XF1 位为“1”时，表明定时计数器 1 工作于高速时钟模式，为“0”时，表明定时计数器 1 工作于系统时钟模式						
4	TC2XF0	TC 高速时钟模式标志位 0，参考定时计数器 0 寄存器描述						
3:2	-	保留						
1	TC2XS1	TC 高速时钟模式选择控制位 1 当设置 TC2XS1 位为“1”时，选择定时计数器 1 工作于高速时钟模式 当设置 TC2XS1 位为“0”时，选择定时计数器 1 工作于系统时钟模式						
0	TC2XS0	TC 高速时钟模式选择控制位 0，参考定时计数器 0 寄存器描述						

TMR0/1/3 预分频器

- 3 个 10 位预分频器
- 复用模式下 TC0、TC1 和 TC3 复用预分频器 CPS310
- 独立模式下 TC0 独用预分频器 CPS310，TC1 独用预分频器 CPS1，TC3 独用预分频器 CPS3
- 支持软件复位

概述

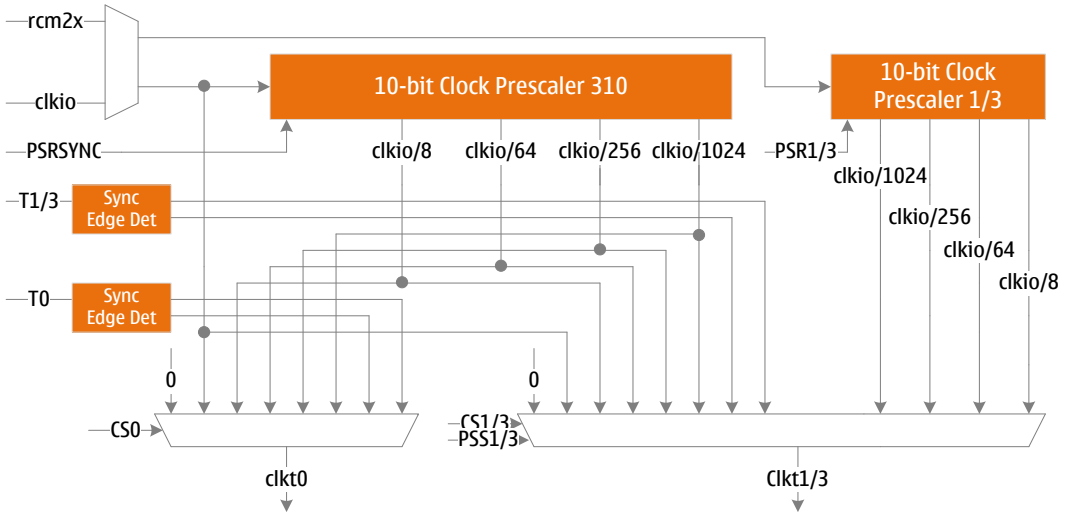
复用模式下（PSS1=0 且 PSS3=0），TC0、TC1 和 TC3 共用一个 10 位的预分频器 CPS310，但它们有不同的分频设置。

单用模式下（PSS1=1 且 PSS3=0），TC1 独立使用一个 10 位的预分频器 CPS1，TC0 和 TC3 共用一个 10 位的预分频器 CPS310，但它们有不同的分频设置。

单用模式下（PSS1=0 且 PSS3=1），TC3 独立使用一个 10 位的预分频器 CPS3，TC0 和 TC1 共用一个 10 位的预分频器 CPS310，但它们有不同的分频设置。

独立模式下（PSS1=1 且 PSS3=1），TC0 独立使用一个 10 位的预分频器 CPS310，TC1 独立使用一个 10 位的预分频器 CPS1，TC3 独立使用预分频器 CPS3。

以下的描述使用于 TC0，TC1 和 TC3，其中 n 代表 0，1 或 3。



TC0/TC1 /TC3 Prescaler 结构图

内部时钟源

当设置 CSn[2:0]=1 时，定时器 3 只可由系统时钟 clkio 驱动，定时计数器 0 或 1 可直接由系统时钟 clkio 或高速时钟 rcm2x（内部 32M RC 振荡器输出时钟的 2 倍频）驱动。预分频器可以输出 4 个不同的时钟频率，分别是 clkio/8，clkio/64，clkio/256 和 clkio/1024。

分频器复位

复用模式

当设置 PSS1 位为“0”且 PSS3 位为“0”时，TC0、TC1 和 TC3 共用一个预分频器 CPS310。

预分频器是独立运行的，其操作独立于 TC 的时钟选择逻辑，且它由 TC0、TC1 和 TC3 共享。由于不受时钟选择控制的影响，预分频器的状态对分频时钟的应用会有影响。当定时器使能并且选用预分频器的输出作为计数时钟源（6>CSn[2:0]>1）时，影响就会产生。从定时器使能到第一次计数可能要花费 1 到 N+1 个系统时钟，其中 N 为预分频因子（8，64，256 或 1024）。

通过复位预分频器来同步定时器和程序运行是可能的。但是必须注意，另一个定时器是否正在使用这个预分频器，复位预分频器会影响到所有与其连接的定时器。

单用模式

当设置 PSS1 位为“1”时，TC1 独立使用预分频器 CPS1，预分频器的复位由 PSR1 位来控制。各自的复位单独起作用，不会影响其它预分频器。

当设置 PSS3 位为“1”时，TC3 独立使用预分频器 CPS3，预分频器的复位由 PSR3 位来控制。各自的复位单独起作用，不会影响其它预分频器。

当设置 PSS1 位为“1”且 PSS3 位为“1”时，TC0 独立使用预分频器 CPS310，预分频器的复位由 PSRSYNC 位来控制，TC1 独立使用预分频器 CPS1，TC3 独立使用预分频器 CPS3，各自的复位单独起作用，不会影响其它预分频器。

外部时钟源

由 T0/T1/T3 引脚提供的外部时钟源可以用作计数时钟源。T0/T1/T3 引脚的信号经过同步逻辑和边沿检测器之后作为计数器的时钟源。每个上升沿（CSn[2:0]=7）或下降沿（CSn[2:0]=6）都会产生一个计数脉冲。外部时钟源不会送入预分频器。

由于引脚上同步与边沿检测电路的存在，T0/T1/T3 上电平的变化需要延迟 2.5 到 3.5 个系统时钟才能使计数器更新。

禁止或使能时钟输入必须在 T0/T1/T3 保持稳定至少需要一个系统时钟周期后才能进行，否则有产生错误计数时钟脉冲的可能。

为了保证正确的采样，外部时钟脉冲宽度必须大于一个系统时钟周期，在占空比为 50% 时外部时钟频率必须小于系统时钟频率的一半。由于振荡器本身的误差带来的系统时钟频率及占空比的差异，建议外部时钟的最高频率不要大于 $f_{sys}/2.5$。

寄存器定义

GTCCR– 通用定时计数器控制寄存器

GTCCR– 通用定时计数器控制寄存器								
地址: 0x43					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	TSM	-	-	-	-	-	PSRASY	PSRSYNC
R/W	R/W	-	-	-	-	-	W	W

Bit	Name	描述
7	TSM	定时计数器同步模式控制位。 当设置 TSM 位为“1”时，为定时计数器同步模式。同步模式下，写入 PSRASY 位和 PSRSYNC 位的值会保持，让相应的预分频器一直被复位。这能确保相应的定时计数器中止并配置成相同的值。 当设置 TSM 位为“0”时，PSRASY 位和 PSRSYNC 位的值会被硬件清零，且定时计数器同时开始工作。
6:2	-	保留。
1	PSRASY	见定时器 TC2 寄存器描述。
0	PSRSYNC	预分频器 CPS310 复位控制位。 当设置 PSRSYNC 位为“1”时，预分频器 CPS310 将被复位。当 TSM 位未置位时，复位之后硬件将清零 PSRSYNC 位。 当设置 PSRSYNC 位为“0”时，设置无效。 复用模式下，TC0/TC1/TC3 共用预分频器，复位将会影响这三个定时器。 独立模式下，复位只会影响 TC0。 读取这一位的值将始终为“0”。

PSSR– 预分频器选择寄存器

PSSR– 预分频器选择寄存器								
地址: 0xE2					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	PSS1	PSS3	-	-	-	-	PSR3	PSR1
R/W	R/W	R/W	-	-	-	-	R/W	R/W

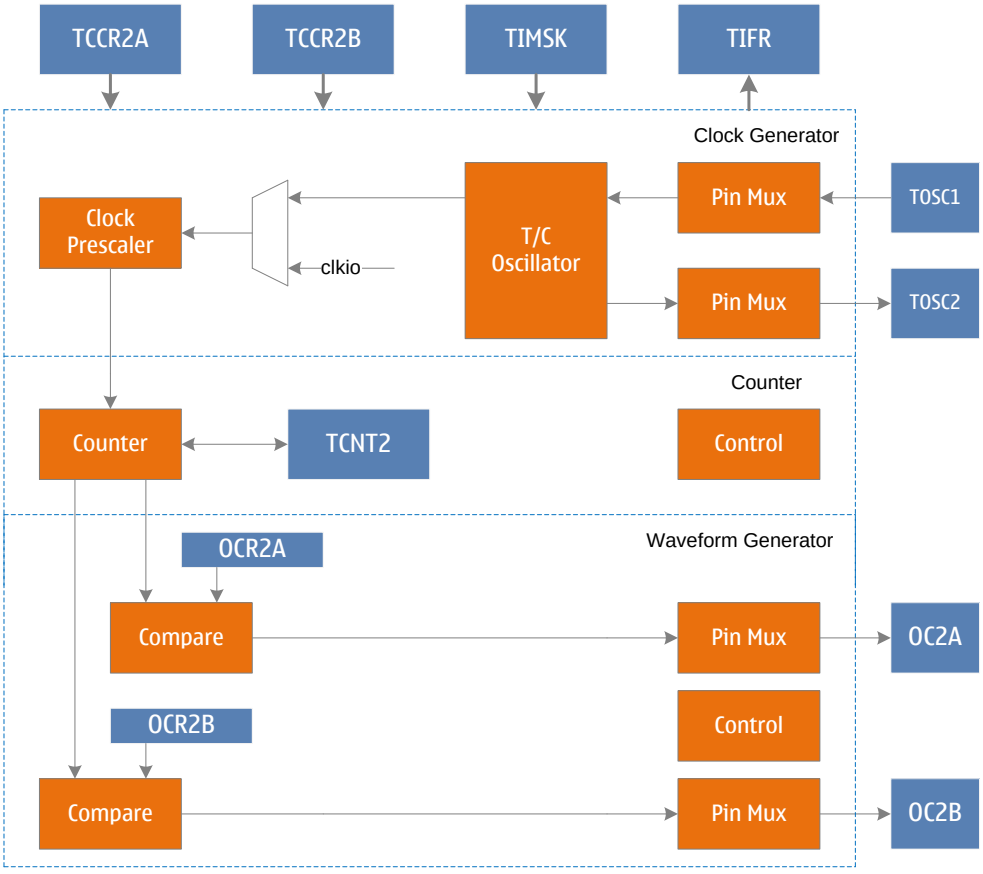
Bit	Name	描述
7	PSS1	预分频器选择控制位。 当设置 PSS1 位为“1”时，TC1 单独使用预分频器 CPS1。 当设置 PSS1 位为“0”时，为预分频器复用模式。TC0 和 TC1 共用预分频器 CPS310。预分频器 CPS1 无效，将会一直被复位。若 PSS3 位同时为“0”，TC3 和 TC0、TC1 共用预分频器 CPS310。预分频器 CPS1 和 CPS3 均无效，将会一直被复位。
6	PSS3	预分频器选择控制位。 当设置 PSS3 位为“1”时，TC3 单独使用预分频器 CPS3。 当设置 PSS3 位为“0”时，为预分频器复用模式。TC0 和 TC3 共用

		预分频器 CPS310。预分频器 CPS3 无效，将会一直被复位。若 PSS1 位同时为“0”，TC1 和 TC0、TC3 共用预分频器 CPS310。预分频器 CPS1 和 CPS3 均无效，将会一直被复位。
5:2	-	保留。
1	PSR3	预分频器 CPS3 复位控制位。 PSR3 位只在 TC3 单用模式下有效。当设置 PSR3 位为“1”时，预分频器 CPS3 将被复位。复位之后硬件将清零 PSR3 位。当设置 PSR3 位为“0”时，设置无效。 读取这一位的值将始终为“0”。
0	PSR1	预分频器 CPS1 复位控制位。 PSR1 位只在 TC1 单用模式下有效。当设置 PSR1 位为“1”时，预分频器 CPS1 将被复位。复位之后硬件将清零 PSR1 位。当设置 PSR1 位为“0”时，设置无效。 读取这一位的值将始终为“0”。

定时/计数器 2 (TMR2)

- 8 位计数器
- 两个独立的比较单元
- 比较匹配发生时自动清零计数器并自动加载
- 无干扰脉冲的相位修正的 PWM 输出
- 频率发生器
- 外部事件计数器
- 10 位的时钟预分频器
- 溢出和比较匹配中断
- 允许使用外部 32.768KHz 的 RTC 晶振计数

概述



TC2 结构图

TC2 是一个通用 8 位定时计数器模块，支持 PWM 输出，可以精确地产生波形。TC2 包含 1 个 8 位计数器，波形产生模式控制单元和 2 个输出比较单元。波形产生模式控制单元控制着计数器的工作模式和比较输出波形的产生。根据不同的工作模式，计数器对每一个计数时钟 Clkt2 实现清零、加一或减一操作。Clkt2 可以由内部时钟源或外部时钟源产生。当使用外部 32.768KHz 的晶振计数时，TC2 可用作 RTC 计数器。当计数器的计数值 TCNT2

到达最大值（等于极大值 0xFF 或输出比较寄存器 OCR2A，定义为 TOP，定义极大值为 MAX 以示区别）时，计数器会进行清零或减一操作。当计数器的计数值 TCNT2 到达最小值（等于 0x00，定义为 BOTTOM）时，计数器会进行加一操作。当计数器的计数值 TCNT2 到达 OCR2A/OCR2B 时，也被称为发生比较匹配时，会清零或置位输出比较信号 OC2A/OCR2B，来产生 PWM 波形。

工作模式

定时计数器 2 有四种不同的工作模式，包括普通模式（Normal），比较匹配时清零（CTC）模式，快速脉冲宽度调制（FPWM）模式和相位修正脉冲宽度调制（PCPWM）模式，由波形产生模式控制位 WGM2[2:0]来选择。下面具体来描述这四种模式。由于有两个独立的输出比较单元，分别用“A”和“B”来表示，用小写的“x”来表示这两个输出比较单元通道。

普通模式

普通模式是定时计数器最简单的工作模式，此时波形产生模式控制位 WGM2[2:0]=0，计数的最大值 TOP 为 MAX（0xFF）。在这种模式下，计数方式为每一个计数时钟加一递增，当计数器到达 TOP 溢出后就回到 BOTTOM 重新开始累加。在计数值 TCNT2 变成零的同一个计数时钟里置位定时计数器溢出标志 TOV2。这种模式下 TOV2 标志就像是第 9 计数位，只是只会被置位不会被清零。溢出中断服务程序会自动清除 TOV2 标志，软件可以用它来提高定时计数器的分辨率。普通模式下没有特殊情形需要考虑，可以随时写入新的计数值。
设置 OC2x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC2x 的波形。当 COM2x=1 时，发生比较匹配时会翻转 OC2x 信号，这种情况下波形的频率可以用下面的公式来计算：

$f_{oc2xnormal} = f_{sys}/(2*N*256)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。
输出比较单元可以用来产生中断，但是在普通模式下不推荐使用中断，这样会占用太多 CPU 的时间。

CTC 模式

设置 WGM2[2:0]=2 时，定时计数器 2 进入 CTC 模式，计数的最大值 TOP 为 OCR2A。在这个模式下，计数方式为每一个计数时钟加一递增，当计数器的数值 TCNT2 等于 TOP 时计数器清零。OCR2A 定义了计数的最大值，亦即计数器的分辨率。这个模式使得用户可以很容易的控制比较匹配输出的频率，也简化了外部事件计数的操作。
当计数器到达计数的最大值时，输出比较匹配标志 OCF2 被置位，相应的中断使能置位时将会产生中断。在中断服务程序里可以更新 OCR2A 寄存器即计数的最大值。在这个模式下 OCR2A 没有使用双缓冲，在计数器以无预分频器或很低的预分频器工作下将最大值更新为接近最小值的时候要小心。如果写入 OCR2A 的数值小于当时的 TCNT2 值时，计数器将丢失一次比较匹配。在下一次比较匹配发生之前，计数器不得不先计数到 TOP，然后再从 BOTTOM 开始计数到 OCR2A 值。和普通模式一样，计数值回到 BOTTOM 的计数时钟里置位 TOV2 标志。
设置 OC2x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC2x 的波形。当 COM2x=1 时，发生比较匹配时会翻转 OC2x 信号，这种情况下波形的频率可以用下面的公式来计算：

$f_{oc2xctc} = f_{sys}/(2*N*(1+OCR2A))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。从公式可以看出，当设置 OCR2x 为 0x0 且无预分频器时，可以获得最大频率为 $f_{sys}/2$ 的输出波形。

快速 PWM 模式

设置 WGM2[2:0]=3 或 7 时，定时计数器 2 进入快速 PWM 模式，可以用来产生高频的 PWM 波形，计数最大值 TOP 分别为 MAX（0xFF）或 OCR2A。快速 PWM 模式和其他 PWM 模式不同在于它是单向操作。计数器从最小值 0x00 累加到 TOP 后又回到 BOTTOM 重新计数。当计数值 TCNT2 到达 OCR2x 或 BOTTOM 时，输出比较信号 OC2x 会被置位或清零，取决于比较输出模式 COM2x 的设置，详情见寄存器描述。由于采用单向操作，快速 PWM 模式的操作频率是采用双向操作的相位修正 PWM 模式的两倍。高频特性使得快速 PWM 模式适用于功率调节，整流以及 DAC 应用。高频信号可以减小外部元器件（电感电容等）的尺寸，从而降低系统成本。

当计数值到达最大值时，定时计数器溢出标志 TOV2 将会被置位，并把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR2x 寄存器。

设置 OC2x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC2x 的波形。波形的频率可用下面的公式来计算：

$f_{oc2xfpwm} = f_{sys}/(N*(1+TOP))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

当 TCNT2 和 OCR2x 发生比较匹配时，波形产生器就置位（清零）OC2x 信号，当 TCNT2 被清零时，波形产生器就清零（置位）OC2x 信号，以此来产生 PWM 波。由此 OCR2x 的极值将会产生特殊的 PWM 波形。当 OCR2x 设置为 0x00 时，输出的 PWM 为每(1+TOP)个计数时钟里有一个窄的尖峰脉冲。当 OCR2x 设置为最大值时，输出的波形为持续的高电平或低电平。

相位修正 PWM 模式

当设置 WGM2[2:0]=1 或 5 时，定时计数器 2 进入相位修正 PWM 模式，计数的最大值 TOP 分别为 MAX（0xFF）或 OCR2A。计数器采用双向操作，由 BOTTOM 递增到 TOP，然后又递减到 BOTTOM，再重复此操作。计数到达 TOP 和 BOTTOM 时均改变计数方向，计数值在 TOP 或 BOTTOM 上均只停留一个计数时钟。在递增或递减过程中，计数值 TCNT2 与 OCR2x 匹配时，输出比较信号 OC2x 将会被清零或置位，取决于比较输出模式 COM2x 的设置。与单向操作相比，双向操作可获得的最大频率要小，但其极好的对称性更适合于电机控制。

相位修正 PWM 模式下，当计数到达 BOTTOM 时置位 TOV2 标志，当计数到达 TOP 时把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR2x 寄存器。

设置 OC2x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC2x 的波形。波形的频率可用下面的公式来计算：

$f_{oc2xpcpwm} = f_{sys}/(N*TOP*2)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

在递增计数过程中，当 TCNT2 与 OCR2x 匹配时，波形产生器就清零（置位）OC2x 信号。在递减计数过程中，当 TCNT2 与 OCR2x 匹配时，波形产生器就置位（清零）OC2x 信号。由此 OCR2x 的极值会产生特殊的 PWM 波。当 OCR2x 设置为最大值或最小值时，OC2x 信号输出会一直保持低电平或高电平。

为了保证输出 PWM 波在最小值两侧的对称性，在没有发生比较匹配时，有两种情况下也会翻转 OC2x 信号。第一种情况是，当 OCR2x 的值由最大值 0xFF 改变为其他数据时。当 OCR2x 为最大值，计数值达到最大时，OC2x 的输出与前面降序计数时比较匹配的结果相同，即保持 OC2x 不变。此时会更新比较值为新的 OCR2x 的值（非 0xFF），OC2x 的值会一直保持，直

到升序计数时发生比较匹配而翻转。此时 OC2x 信号并不以最小值为中心对称，因此需要在 TCNT2 到达最大值时翻转 OC2x 信号，此即没有发生比较匹配时翻转 OC2x 信号的第一种情况。第二种情况是，当 TCNT2 从比 OCR2x 高的值开始计数时，因而会丢失一次比较匹配，从而引起不对称情形的产生。同样需要翻转 OC2x 信号去实现最小值两侧的对称性。

TC2 的异步操作方式

当位于 ASSR 寄存器的 AS2 位为“1”时，TC2 工作在异步模式，计数器的时钟源来自于外部定时计数器的振荡器。异步模式下 TC2 的操作要考虑如下几点。

- 在同步和异步模式之间的转换有可能造成 TCNT2、OCR2A、OCR2B、TCCR2A 和 TCCR2B 数据的损坏。安全的操作步骤如下所示：
  1. 清零 OCIE2A，TOIE2 和 OCIE2B 寄存器位来关闭 TC2 的中断；
  2. 置位 AS2 位选择合适的时钟源；
  3. 对 TCNT2、OCR2A、TCCR2A、OCR2B 和 TCCR2B 寄存器写入新的数据；
  4. 切换到异步模式时，需等待 TCN2UB、OCR2AUB、TCR2AUB、OCR2BUB 和 TCR2BUB 位清零；
  5. 清零 TC2 的中断标志位；
  6. 使能需要使用的中断。
- 振荡器最好使用 32.768KHz 的手表晶振。系统时钟频率必须比晶振频率高 4 倍以上。
- CPU 写 TCNT2、OCR2A、TCCR2A、OCR2B 和 TCCR2B 时，硬件会将数据先放入暂存器，两个 TOSC1 时钟上升沿后才锁存到对应的寄存器中。在数据从暂存器锁存到目的寄存器之前不能执行新的数据写入操作。各个寄存器都有各自独立的暂存器，因此写 TCNT2 并不会干扰写 OCR2。异步状态寄存器 ASSR 用来检查数据是否已经写入到目的寄存器。
- 如果使用 TC2 作为 MCU 休眠模式的唤醒条件，则在各个寄存器更新结束之前不能进入休眠模式，否则 MCU 可能会在 TC2 设置生效之前进入休眠模式，从而 TC2 无法唤醒系统。
- 如果使用 TC2 作为 MCU 休眠模式的唤醒条件，必须注意重新进入休眠模式的过程。中断逻辑需要一个 TOSC1 时钟周期进行复位，如果从唤醒到重新进入休眠的时间小于一个 TOSC1 时钟周期，中断将不再发生，器件也无法唤醒。推荐采用如下的操作方法：
  1. 对各个寄存器写入合适的数据；
  2. 等待 ASSR 相应的更新忙标志位清零；
  3. 进入休眠模式。
- 若选择了异步工作模式，TC2 的振荡器将一直工作，除非进入掉电模式。用户必须注意，此振荡器的稳定时间可能长达 1 秒钟，因此，建议用户在使能 TC2 的振荡器后至少等待 1 秒钟后再使用 TC2 的异步工作模式。
- 异步工作模式时休眠模式下唤醒的过程：中断条件满足后，在下一个定时器时钟启动唤醒过程。也就是说，在处理器可以读取计数器的数值之前计数器至少又累加了一个时钟。唤醒后 MCU 执行中断服务程序，之后开始执行 SLEEP 语句之后的程序。
- 从休眠模式唤醒之后短时间内读取 TCNT2 的值可能返回不正确的数据。因为 TCNT2 是由异步的 TOSC1 时钟驱动的，而读取 TCNT2 必须通过一个内部系统时钟同步的寄存器来完成，同步发生于每个 TOSC1 的上升沿。从休眠模式唤醒后系统时钟重新激活，读取的 TCNT2 数值为进入休眠模式之前的值，直到下一个 TOSC1 上升沿的到来才会更新。从休眠模式唤醒时 TOSC1 的相位完全不可预测，而与唤醒时间有关。因此，读取 TCNT2 值的推荐序列为：
  1. 写一个任意数值到 OCR2A 或 TCCR2A；

2. 等待相应的更新忙标志位被清零；
3. 读取 TCNT2。

- 异步模式下，中断标志位的同步需要 3 个系统时钟周期加 1 个定时器周期。在 MCU 可以读取引起中断标志置位的计数器数值之前计数器至少又累加了一个时钟。输出比较信号的变化与定时器时钟同步，而不是系统时钟。

TC2 的预分频器

TC2 预分频器的输入时钟称为 clkt2s，由位于 ASSR 寄存器的 AS2 位来选择内部系统时钟 clkio 或者外部 TOSC1 时钟源，缺省为与系统时钟 clkio 相连接。若 AS2 置位，TC2 将由 TOSC1 异步驱动。当 TOSC1 引脚和 TOSC2 引脚外接一个 32.768KHz 的钟表晶振，TC2 可用作 RTC 计数器。不推荐在 TOSC1 引脚上直接施加外部时钟信号。

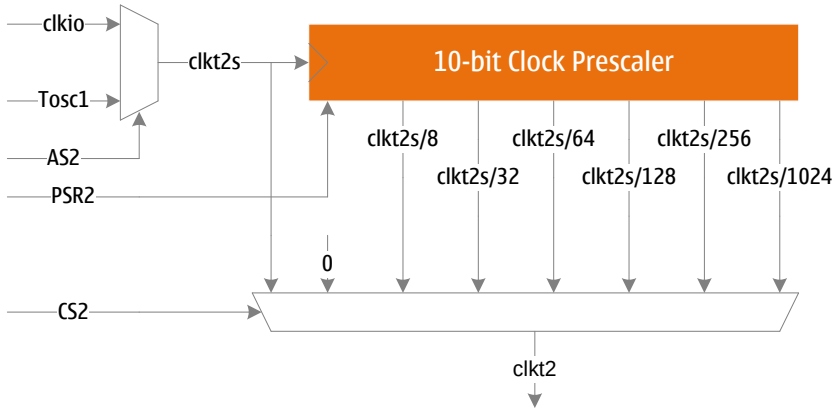


Figure 5 TC2 预分频器结构图

上图为 TC2 预分频器，如图所示，可能的预分频选项有：clkt2s/8, clkt2s/32, clkt2s/64, clkt2s/128, clkt2s/256,和 clkt2s/1024。此外还可以选择 clkt2s 和 0（停止计数）。置位 SFIOR 寄存器的 PSR2 位将复位预分频器，从而允许用户从可预测的预分频器开始工作。

寄存器定义

TC2 寄存器列表

寄存器	地址	默认值	描述
TCCR2A	0xB0	0x00	TC2 控制寄存器 A
TCCR2B	0xB1	0x00	TC2 控制寄存器 B
TCNT2	0xB2	0x00	TC2 计数值寄存器
OCR2A	0xB3	0x00	TC2 输出比较寄存器 A
OCR2B	0xB4	0x00	TC2 输出比较寄存器 B
ASSR	0xB6	0x00	TC2 异步状态寄存器
TIMSK2	0x70	0x00	定时计数器中断屏蔽寄存器
TIFR2	0x37	0x00	定时计数器中断标志寄存器

TCCR2A–TC2 控制寄存器 A

TCCR2 A–TC2 控制寄存器 A								
地址: 0xB0					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	COM2A1	COM2A0	COM2B1	COM2B0	-	-	WGM21	WGM20
R/W	W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	描述
7	COM2A1	TC2 比较匹配输出 A 模式控制高位。 COM2A1 和 COM2A0 一起组成输出比较模式控制 COM2A[1:0], 控制 OC2A 的输出波形。如果 COM2A 的 1 位或者 2 位都置位，输出比较波形占据着 OC2A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM2A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
6	COM2A0	TC2 比较匹配输出 A 模式控制低位。 COM2A0 和 COM2A1 一起组成输出比较模式控制 COM2A[1:0], 控制 OC2A 的输出波形。如果 COM2A 的 1 位或者 2 位都置位，输出比较波形占据着 OC2A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM2A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
5	COM2B1	TC2 比较匹配输出 B 模式控制高位。 COM2B1 和 COM2B0 一起组成输出比较模式控制 COM2B[1:0], 控制 OC2B 的输出波形。如果 COM2B 的 1 位或者 2 位都置位，输出比较波形占据着 OC2B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM2B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
4	COM2B0	TC2 比较匹配输出 B 模式控制低位。 COM2B0 和 COM2B1 一起组成输出比较模式控制 COM2B[1:0], 控制 OC2B 的输出波形。如果 COM2B 的 1 位或者 2 位都置位，输出比较波形占据着 OC2B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM2B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
3:2	-	保留。
1	WGM21	TC2 波形产生模式控制高位。 WGM20 和 WGM21，WGM22 一起组成波形产生模式控制 WGM2[2:0], 控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
0	WGM20	TC2 波形产生模式控制低位。 WGM21 和 WGM20，WGM22 一起组成波形产生模式控制 WGM2[2:0], 控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。

TCCR2B –TC2 控制寄存器 B

TCCR2B –TC2 控制寄存器 B								
地址: 0xB1					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	FOC2A	FOC2B	-	-	WGM22	CS22	CS21	CS20
R/W	W	W	-	-	R/W	R/W	R/W	R/W

Bit	Name	描述
7	FOC2A	TC2 强制输出比较 A 控制位。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC2A 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF2A 标志，也不会重载或清零定时器，但是输出引脚 OC2A 将被按照 COM2A 的设置相应的更新，就跟真的发生了比较匹配一样。 读取 FOC2A 的返回值一直为零。
6	FOC2B	TC2 强制输出比较 B 控制位。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC2B 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF2B 标志，也不会重载或清零定时器，但是输出引脚 OC2B 将被按照 COM2B 的设置相应的更新，就跟真的发生了比较匹配一样。 读取 FOC2B 的返回值一直为零。
5:4	-	保留。
3	WGM22	TC2 波形产生模式控制高位。 WGM22 和 WGM20, WGM21 一起组成波形产生模式控制 WGM2[2:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
2	CS22	TC2 时钟选择控制高位。 用于选择定时计数器 2 的时钟源。
1	CS21	TC2 时钟选择控制中位。 用于选择定时计数器 2 的时钟源。
0	CS20	TC2 时钟选择控制低位。 用于选择定时计数器 2 的时钟源。

CS2[2:0]	描述
0	无时钟源，停止计数
1	clk_{t2s}
2	clk_{t2s}/8，来自预分频器
3	clk_{t2s}/32，来自预分频器
4	clk_{t2s}/64，来自预分频器
5	clk_{t2s}/128，来自预分频器
6	clk_{t2s}/256，来自预分频器
7	clk_{t2s}/1024，来自预分频器

下表为非 PWM 模式（即普通模式和 CTC 模式）下，比较输出模式对输出比较波形的控制。

Table 1 非 PWM 模式下 OC2x 比较输出模式控制

COM2x[1:0]	描述
0	OC2x 断开，通用 IO 口操作
1	比较匹配时翻转 OC2x 信号
2	比较匹配时清零 OC2x 信号
3	比较匹配时置位 OC2x 信号

下表为快速 PWM 模式下比较输出模式对输出比较波形的控制。

Table 2 快速 PWM 模式下 OC2x 比较输出模式控制

COM2x[1:0]	描述
0	OC2x 断开，通用 IO 口操作
1	保留
2	比较匹配时清零 OC2x 信号，最大值匹配时置位 OC2x 信号
3	比较匹配时置位 OC2x 信号，最大值匹配时清零 OC2x 信号

下表为相位修正模式下比较输出模式对输出比较波形的控制。

Table 3 相位修正 PWM 模式下 OC2x 比较输出模式控制

COM2x[1:0]	描述
0	OC2x 断开，通用 IO 口操作
1	保留
2	升序计数下比较匹配时清零 OC2x 信号，降序计数下比较匹配时置位 OC2x 信号
3	升序计数下比较匹配时置位 OC2x 信号，降序计数下比较匹配时清零 OC2x 信号

下表为波形产生模式控制。

Table 4 波形产生模式控制

WGM2[2:0]	工作模式	TOP 值	更新 OCR2x 时刻	置位 TOV2 时刻
0	Normal	0xFF	立即	MAX
1	PCPWM	0xFF	TOP	BOTTOM
2	CTC	OCR2A	立即	MAX
3	FPWM	0xFF	TOP	MAX
4	保留	-	-	-
5	PCPWM	OCR2A	TOP	BOTTOM
6	保留	-	-	-
7	FPWM	OCR2A	TOP	TOP

TCNT2 –TC2 计数值寄存器

TCNT2 –TC2 计数值寄存器								
地址: 0xB2					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	TCNT27	TCNT26	TCNT25	TCNT24	TCNT23	TCNT22	TCNT21	TCNT20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	TCNT2	TC2 计数值寄存器。 通过 TCNT2 寄存器可以直接对计数器的 8 为计数值进行读写访问。 CPU 对 TCNT2 寄存器的写操作会在下一个定时器时钟周期阻止比较匹配的发生，即使定时器已经停止。这就允许初始化 TCNT2 寄存器的值与 OCR2 的值一致而不会引发中断。 如果写入 TCNT2 的数值等于或绕过 OCR2 值时，比较匹配就会丢失，造成不正确的波形发生结果。 没有选择时钟源时定时器停止计数，但 CPU 仍可以访问 TCNT2。CPU 写计数器比清零或加减操作的优先级高。

OCR2A – TC2 输出比较寄存器 A

OCR2A – TC2 输出比较寄存器 A								
地址: 0xB3					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR2A7	OCR2A6	OCR2A5	OCR2A4	OCR2A3	OCR2A2	OCR2A1	OCR2A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	OCR2A	TC2 输出比较寄存器 A。 OCR2A 包含一个 8 位的数据，不间断地与计数器数值 TCNT2 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC2A 引脚上产生波形。 当使用 PWM 模式时，OCR2A 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR2A 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR2A 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR2A 本身。

OCR2B – TC2 输出比较寄存器 B

OCR2B – TC2 输出比较寄存器 B								
地址: 0xB4					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR2B7	OCR2B6	OCR2B5	OCR2B4	OCR2B3	OCR2B2	OCR2B1	OCR2B0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	OCR2B	TC2 输出比较 B 寄存器。 OCR2B 包含一个 8 位的数据，不间断地与计数器数值 TCNT2 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC2B 引脚上产生波形。 当使用 PWM 模式时，OCR2B 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR2B 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。使用双缓冲功能时，CPU 访问的是 OCR2B 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR2B 本身。

TIMSK2 – TC2 中断屏蔽寄存器

TIMSK2 – TC2 中断屏蔽寄存器								
地址: 0x70					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCIE2B	OCIE2A	TOIE2
R/W	-	-	-	-	-	R/W	R/W	R/W

Bit	Name	描述
7:3		保留。
2	OCIE2B	TC2 输出比较 B 匹配中断使能位。 当 OCIE2B 位为“1”，且全局中断置位，TC2 输出比较 B 匹配中断使能。当比较匹配发生时，即 TIFR2 中 OCF2B 位被置位时，中断产生。 当 OCIE2B 位为“0”时，TC2 输出比较 B 匹配中断被禁止。
1	OCIE2A	TC2 输出比较 A 匹配中断使能位。 当 OCIE2A 位为“1”，且全局中断置位，TC2 输出比较 A 匹配中断使能。当比较匹配发生时，即 TIFR2 中 OCF2A 位被置位时，中断产生。 当 OCIE2A 位为“0”时，TC2 输出比较 A 匹配中断被禁止。
0	TOIE2	TC2 溢出中断使能位。 当 TOIE2 位为“1”，且全局中断置位，TC2 溢出中断使能。当 TC2 发生溢出，即 TIFR2 中的 TOV2 位被置位时，中断产生。当 TOIE2 位为“0”时，TC2 溢出中断被禁止。

TIFR2 – TC2 中断标志寄存器

TIFR2 – TC2 中断标志寄存器								
地址: 0x37					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	OCF2B	OCF2A	TOV2
R/W	-	-	-	-	-	R/W	R/W	R/W

Bit	Name	描述
7:3	-	保留。
2	OCF2B	TC2 输出比较 B 匹配标志位。

		当 TCNT2 等于 OCR2B 时，比较单元就给出匹配信号，并置位比较标志 OCF2B。若此时输出比较 B 中断使能 OCIE2B 为“1”且全局中断标志置位，则会产生输出比较 B 中断。执行此中断服务程序时 OCF2B 将自动清零，或对 OCF2B 位写“1”也可清零该位。
1	OCF2A	TC2 输出比较 A 匹配标志位。 当 TCNT2 等于 OCR2A 时，比较单元就给出匹配信号，并置位比较标志 OCF2A。若此时输出比较 A 中断使能 OCIE2A 为“1”且全局中断标志置位，则会产生输出比较 A 中断。执行此中断服务程序时 OCF2A 将自动清零，或对 OCF2A 位写“1”也可清零该位。
0	TOV2	TC2 溢出标志位。 当计数器发生溢出时，置位溢出标志 TOV2。若此时溢出中断使能 TOIE2 为“1”且全局中断标志置位，则会产生溢出中断。执行此中断服务程序时 TOV2 将自动清零，或对 TOV2 位写“1”也可清零该位。

ASSR – 异步接口状态寄存器

ASSR– TC2 异步接口状态寄存器								
地址: 0xB6					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	INTCK	-	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB
R/W	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	INTCK	异步时钟选择控制位。 当设置 INTCK 位为 1 时，选择内部 RC32K 作为异步时钟源。 当设置 INTCK 位为 0 时，选择外部晶振时钟作为异步时钟源。
6	-	保留。
5	AS2	定时器 2 异步模式选择控制位。 当设置 AS2 位为 1 时，定时器 2 工作为异步模式，其时钟源由 INTCK 位来选择。 当设置 AS2 位为 0 时，定时器 2 工作为同步模式，其时钟源为 Clk_{io}。 当 AS2 的值发生改变时，TCNT2，OCR2A，OCR2B，TCCR2A 和 TCCR2B 寄存器的值可能会不正确，需重新配置。
4	TCN2UB	TCNT2 寄存器更新标志位。 当定时器 2 工作于异步模式下，对 TCNT2 进行写入操作时，TCN2UB 位会被置位。当 TCNT2 的值更新完毕后，硬件会清零 TCN2UB 位。只有当 TCN2UB 位为 0 时，才可对 TCNT2 进行更新。
3	OCR2AUB	OCR2A 寄存器更新标志位。 当定时器 2 工作于异步模式下，对 OCR2A 进行写入操作时，OCR2AUB 位会被置位。当 OCR2A 的值更新完毕后，硬件会清零 OCR2AUB 位。只有当 OCR2AUB 位为 0 时，才可对 OCR2A 进行更新。
2	OCR2BUB	OCR2B 寄存器更新标志位。 当定时器 2 工作于异步模式下，对 OCR2B 进行写入操作时，OCR2BUB 位会被置位。当 OCR2B 的值更新完毕后，硬件会清零 OCR2BUB 位。只有当 OCR2BUB 位为 0 时，才可对 OCR2B 进行更新。

1	TCR2AUB	TCCR2A 寄存器更新标志位。 当定时器 2 工作于异步模式下，对 TCCR2A 进行写入操作时，TCR2AUB 位会被置位。当 TCCR2A 的值更新完毕后，硬件会清零 TCR2AUB 位。只有当 TCR2AUB 位为 0 时，才可对 TCCR2A 进行更新。
0	TCR2BUB	TCCR2B 寄存器更新标志位。 当定时器 2 工作于异步模式下，对 TCCR2B 进行写入操作时，TCR2BUB 位会被置位。当 TCCR2B 的值更新完毕后，硬件会清零 TCR2BUB 位。只有当 TCR2BUB 位为 0 时，才可对 TCCR2B 进行更新。

定时/计数器 3 (TMR3)

- 真正的 16 位设计，允许 16 位的 PWM
- 3 个独立的输出比较单元
- 双缓冲的输出比较寄存器
- 1 个输入捕捉单元
- 输入捕捉噪声抑制器
- 比较匹配时自动清零计数器并自动加载
- 无干扰脉冲的相位修正的 PWM
- 可变的 PWM 周期
- 频率发生器
- 外部事件计数器
- 5 个独立的中断源
- 带死区时间控制
- 6 个可选触发源自动关闭 PWM 输出

概述

TC3 是一个通用 16 位定时计数器模块，支持 PWM 输出，可以精确地产生波形。TC3 包含 1 个 16 位计数器，波形产生模式控制单元，2 个独立的输出比较单元和 1 个输入捕捉单元。波形产生模式控制单元控制着计数器的工作模式和比较输出波形的产生。根据不同的工作模式，计数器对每一个计数时钟 Clkt3 实现清零、加一或减一操作。Clkt3 可以由内部时钟源或外部时钟源产生。当计数器的计数值 TCNT3 到达最大值（等于极大值 0xFFFF 或固定值或输出比较寄存器 OCR3A 或输入捕捉寄存器 ICR3，定义为 TOP，定义极大值为 MAX 以示区别）时，计数器会进行清零或减一操作。当计数器的计数值 TCNT3 到达最小值（等于 0x0000，定义为 BOTTOM）时，计数器会进行加一操作。当计数器的计数值 TCNT3 到达 OCR3A 或 OCR3B 或 OCR3C 时，也被称为发生比较匹配时，会清零或置位输出比较信号 OC3A 或 OC3B 或 OC3C，来产生 PWM 波形。当开启输入捕捉功能时，计数器被触发即开始或停止计数，ICR3 寄存器会记录捕捉信号触发周期内的计数值。

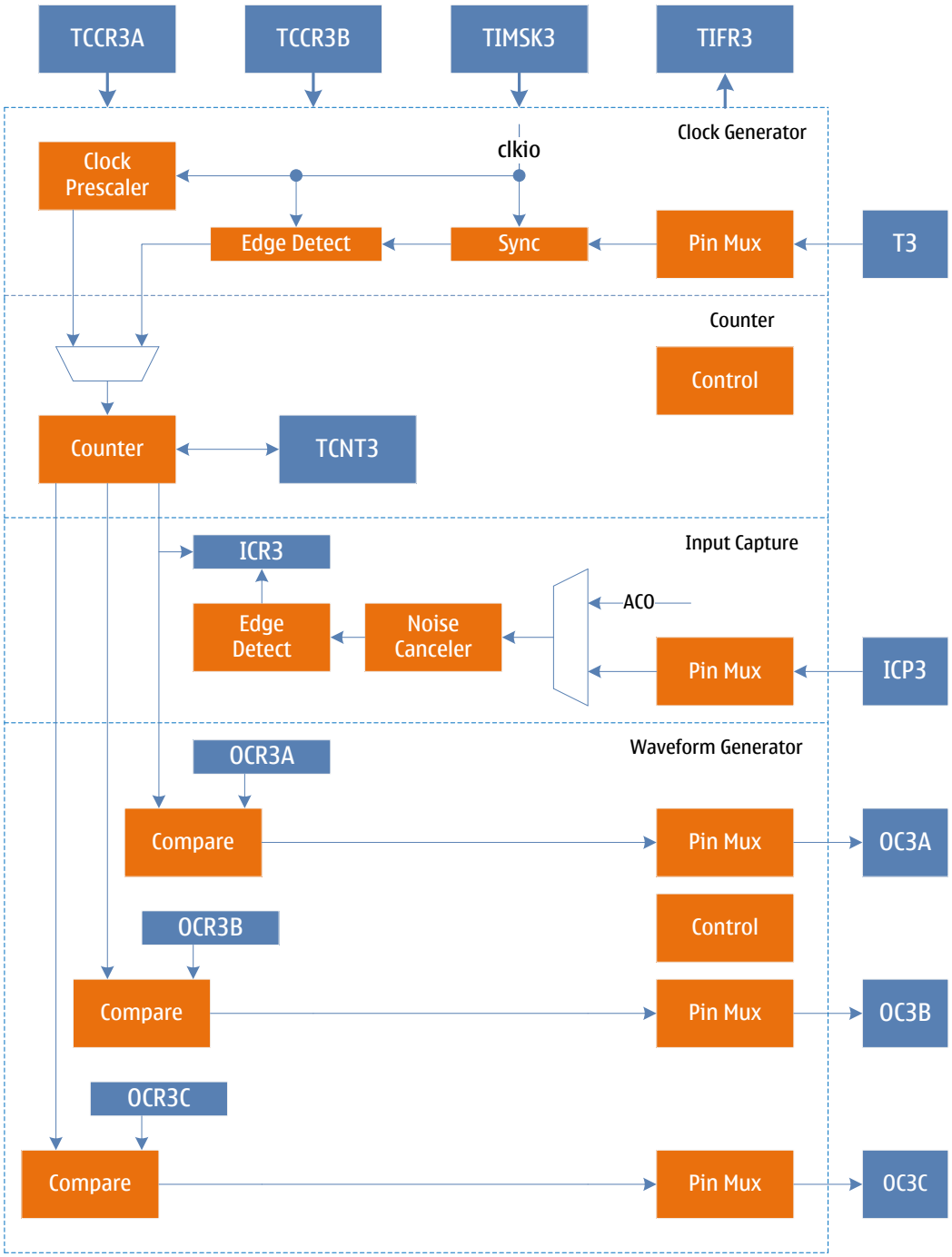


Figure 6 TC3 结构图

工作模式

定时计数器 1 有六种不同的工作模式，包括普通模式（Normal），比较匹配时清零（CTC）模式，快速脉冲宽度调制（FPWM）模式，相位修正脉冲宽度调制（PCPWM）模式，相位频率修正脉冲宽度调制（PFCPWM）模式，和输入捕捉（ICP）模式。由波形产生模式控制位 WGM3[3:0]来选择。下面具体来描述这六种模式。由于有三个独立的输出比较单元，分别用“A”、“B”和“C”来表示，用小写的“x”来表示这两个输出比较单元通道。

普通模式

普通模式是定时计数器最简单的工作模式，此时波形产生模式控制位 WGM3[3:0]=0，计数的最大值 TOP 为 MAX（0xFFFF）。在这种模式下，计数方式为每一个计数时钟加一递增，当计数器到达 TOP 溢出后就回到 BOTTOM 重新开始累加。在计数值 TCNT3 变成零的同一个计数时钟里置位定时计数器溢出标志 TOV3。这种模式下 TOV3 标志就像是第 17 计数位，只是只会被置位不会被清零。溢出中断服务程序会自动清除 TOV3 标志，软件可以用它来提高定时计数器的分辨率。普通模式下没有特殊情形需要考虑，可以随时写入新的计数值。

设置 OC3x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC3x 的波形。当 COM3x=1 时，发生比较匹配时会翻转 OC3x 信号，这种情况下波形的频率可以用下面的公式来计算：

$f_{OC3xnormal} = f_{sys}/(2*N*65536)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

输出比较单元可以用来产生中断，但是在普通模式下不推荐使用中断，这样会占用太多 CPU 的时间。

CTC 模式

设置 WGM3[3:0]=4 或 12 时，定时计数器 1 进入 CTC 模式。当 WGM3[3]=0 时，计数最大值 TOP 为 OCR3A，当 WGM3[3]=1 时，计数最大值 TOP 为 ICR3。下面以 WGM3[3:0]=4 为例来描述 CTC 模式在这个模式下，计数方式为每一个计数时钟加一递增，当计数器的数值 TCNT3 等于 TOP 时计数器清零。这个模式使得用户可以很容易的控制比较匹配输出的频率，也简化了外部事件计数的操作。

当计数器到达 TOP=OCR3A 时，输出比较匹配标志 OCF3A 被置位，当计数器到达 TOP=ICR3 时，输出比较匹配标志 ICF3 被置位，相应的中断使能置位时将会产生中断。在中断服务程序里可以更新 OCR3A 寄存器。在这个模式下 OCR3A 没有使用双缓冲，在计数器以无预分频器或很低的预分频器工作下将最大值更新为接近最小值的时候要小心。如果写入 OCR3A 的数值小于当时的 TCNT3 值时，计数器将丢失一次比较匹配。在下一次比较匹配发生之前，计数器不得不先计数到 MAX，然后再从 BOTTOM 开始计数到 OCR3A。和普通模式一样，计数值回到 0x0 的计数时钟里置位 TOV3 标志。

设置 OC3x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC3x 的波形。波形的频率可以用下面的公式来计算：

$f_{OC3xctc} = f_{sys}/(2*N*(1+OCR3A))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

从公式可以看出，当设置 OCR3A 为 0x0 且无预分频器时，可以获得最大频率为 $f_{sys}/2$ 的输出波形。

当 WGM3[3:0]=12 时与 WGM3[3:0]=4 类似，只是把与 OCR3A 相关的换成 ICR3 即可。

快速 PWM 模式

设置 WGM3[3:0]=5，6，7，14 或 15 时，定时计数器 1 进入快速 PWM 模式，计数最大值 TOP 分别为 0xFF，0x1FF，0x3FF，ICR3 或 OCR3A，可以用来产生高频的 PWM 波形。快速 PWM

模式和其他 PWM 模式不同在于它是单向操作。计数器从 BOTTOM 累加到 TOP 后又回到 BOTTOM 重新计数。当计数值 TCNT3 到达 TOP 或 BOTTOM 时，输出比较信号 OC3x 会被置位或清零，取决于比较输出模式 COM3 的设置，详情见寄存器描述。由于采用单向操作，快速 PWM 模式的操作频率是采用双向操作的相位修正 PWM 模式的两倍。高频特性使得快速 PWM 模式适用于功率调节，整流以及 DAC 应用。高频信号可以减小外部元器件（电感电容等）的尺寸，从而降低系统成本。

当计数值到达 TOP 时，定时计数器溢出标志 TOV3 将会被置位，并把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新 OCR3A 寄存器。

设置 OC3x 引脚的数据方向寄存器为输出时才能得到输出比较信号 OC3x 的波形。波形的频率可用下面的公式来计算：

$f_{OC3xfpwm} = f_{sys}/(N*(1+TOP))$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

当 TCNT3 和 OCR3x 发生比较匹配时，波形产生器就置位（清零）OC3x 信号，当 TCNT3 被清零时，波形产生器就清零（置位）OC3x 信号，以此来产生 PWM 波。由此 OCR3x 的极值将会产生特殊的 PWM 波形。当 OCR3x 设置为 0x00 时，输出的 PWM 为每(1+TOP)个计数时钟里有一个窄的尖峰脉冲。当 OCR3x 设置为 TOP 时，输出的波形为持续的高电平或低电平。如果用 OCR3A 作为 TOP 并设置 COM3A=1，输出比较信号 OC3A 会产生占空比为 50%的 PWM 波。

相位修正 PWM 模式

当设置 WGM3[3:0]=1，2，3，10 或 11 时，定时计数器 1 进入相位修正 PWM 模式，计数的最大值 TOP 分别为 0xFF，0x1FF，0x3FF，ICR3 或 OCR3A。计数器采用双向操作，由 BOTTOM 递增到 TOP，然后又递减到 BOTTOM，再重复此操作。计数到达 TOP 和 BOTTOM 时均改变计数方向，计数值在 TOP 或 BOTTOM 上均只停留一个计数时钟。在递增或递减过程中，计数值 TCNT3 与 OCR3x 匹配时，输出比较信号 OC3x 将会被清零或置位，取决于比较输出模式 COM3 的设置。与单向操作相比，双向操作可获得的最大频率要小，但其极好的对称性更适合于电机控制。

相位修正 PWM 模式下，当计数到达 BOTTOM 时置位 TOV3 标志，当计数到达 TOP 时把比较缓冲器的值更新到比较值。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR3x 存器。

设置 OC3x 脚的数据方向寄存器为输出时才能得到输出比较信号 OC3x 波形。波形的频率可用下面的公式来计算：

$f_{OC3xcpcpwm} = f_{sys}/(N*TOP*2)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

在递增计数过程中，当 TCNT3 与 OCR3x 匹配时，波形产生器就清零（置位）OC3x 信号。在递减计数过程中，当 TCNT3 与 OCR3x 匹配时，波形产生器就置位（清零）OC3x 信号。由此 OCR3x 的极值会产生特殊的 PWM 波。当 OCR3x 设置为 TOP 或 BOTTOM 时，OC3x 信号输

出会一直保持低电平或高电平。如果用 OCR3A 作为 TOP 并设置 COM3A=1，输出比较信号 OC3A 会产生占空比为 50%的 PWM 波。

为了保证输出 PWM 波在 BOTTOM 两侧的对称性，在没有发生比较匹配时，有两种情况下也会翻转 OC3x 信号。第一种情况是，当 OCR3x 的值由 TOP 改变为其他数据时。当 OCR3x 为 TOP，计数值达到 TOP 时，OC3x 的输出与前面降序计数时比较匹配的结果相同，即保持 OC3x 不变。此时会更新比较值为新的 OCR3x 的值（非 TOP），OC3x 的值会一直保持，直到升序计数时发生比较匹配而翻转。此时 OC3x 信号并不以最小值为中心对称，因此需要在 TCNT3 到达最大值时翻转 OC3x 信号，此即没有发生比较匹配时翻转 OC3x 信号的第一种情况。第二种情况是，当 TCNT3 从比 OCR3x 高的值开始计数时，因而会丢失一次比较匹配，从而引起不对称情形的产生。同样需要翻转 OC3x 信号去实现最小值两侧的对称性。

相位频率修正 PWM 模式

当设置 WGM3[3:0]=8 或 9 时，定时计数器 1 进入相位频率修正 PWM 模式，计数的最大值 TOP 分别为 ICR3 或 OCR3A。计数器采用双向操作，由 BOTTOM 递增到 TOP，然后又递减到 BOTTOM，再重复此操作。计数到达 TOP 和 BOTTOM 时均改变计数方向，计数值在 TOP 或 BOTTOM 上均只停留一个计数时钟。在递增或递减过程中，计数值 TCNT3 与 OCR3x 匹配时，输出比较信号 OC3x 将会被清零或置位，取决于比较输出模式 COM3 的设置。与单向操作相比，双向操作可获得的最大频率要小，但其极好的对称性更适合于电机控制。

相位频率修正 PWM 模式下，当计数到达 BOTTOM 时置位 TOV3 标志，并且把比较缓冲器的值更新到比较值，更新比较值的时间是相位频率修正 PWM 模式和相位修正 PWM 模式的最大不同点。如果中断使能，在中断服务程序中可以更新比较缓冲器 OCR3x 存器。当 CPU 改变 TOP 值即 OCR3A 或 ICR3 的值时，必须保证新的 TOP 值不小于已经在使用的 TOP 值，否则比较匹配将不会再发生。

设置 OC3x 脚的数据方向寄存器为输出时才能得到输出比较信号 OC3x 波形。波形的频率可用下面的公式来计算：

$f_{OC3xcpfcpwm} = f_{sys}/(N*TOP*2)$

其中，N 表示的是预分频因子（1，8，64，256 或者 1024）。

在递增计数过程中，当 TCNT3 与 OCR3x 匹配时，波形产生器就清零（置位）OC3x 信号。在递减计数过程中，当 TCNT3 与 OCR3x 匹配时，波形产生器就置位（清零）OC3x 信号。由此 OCR3x 的极值会产生特殊的 PWM 波。当 OCR3x 设置为 TOP 或 BOTTOM 时，OC3x 信号输出会一直保持低电平或高电平。如果用 OCR3A 作为 TOP 并设置 COM3A=1，输出比较信号 OC3A 会产生占空比为 50%的 PWM 波。

因为 OCR3x 寄存器是在 BOTTOM 时刻更新的，所以 TOP 值两边升序和降序的计数长度是一样的，也就产生了频率和相位都正确的对称波形。

当使用固定 TOP 值时，最好采用 ICR3 寄存器作为 TOP 值，即设置 WGM3[3:0]=8，此时 OCR3A 寄存器只需用来产生 PWM 输出。如果要产生频率变化的 PWM 波，必须通过改变 TOP 值， OCR3A 的双缓冲特性会更适合于这个应用。

输入捕捉模式

输入捕捉用来捕获外部事件，并为其赋予时间标记以说明此事件发生的时刻，可以在前面的计数模式下进行，不过要除去使用 ICR3 值作为计数 TOP 值的波形产生模式。

外部事件发生的触发信号由引脚 ICP3 输入，也可以通过模拟比较器单元来实现。当引脚 ICP3 上的逻辑电平发生变化，或模拟比较器的输出 ACO 电平发生变化，并且这个电平变化被输入捕捉单元所捕获，输入捕捉即被触发，此时 16 位的计数值 TCNT3 数据被复制到输入捕捉寄存器 ICR3，同时输入捕捉标志 ICF3 置位，若 ICIE1 位为“1”，输入捕捉标志将产生输入捕捉中断。

通过设置模拟比较控制与状态寄存器 ACSR 的模拟比较输入捕捉控制位 ACIC 来选择输入捕捉触发源 ICP3 或 ACO。需注意的是，改变触发源有可能造成一次输入捕捉，因此在改变触发源后必须对 ICF3 进行一次清零操作来避免出现错误的结果。

输入捕捉信号经过一个可选的噪声抑制器之后送入边沿检测器，根据输入捕捉选择控制位 ICES1 的配置，看检测到的边沿是否满足触发条件。噪声抑制器是一个简单的数字滤波，对输入信号进行 4 次采样，只有当 4 次采样值都相等时其输出才会送入边沿检测器。噪声抑制器由 TCCR3B 寄存器的 ICNC1 位控制其使能或禁止。

使用输入捕捉功能时，当 ICF3 被置位后，应尽可能早的读取 ICR3 寄存器的值，因为下一次捕捉事件发生后 ICR3 的值将会被更新。推荐使能输入捕捉中断，在任何输入捕捉工作模式下，都不推荐在操作过程中改变计数 TOP 值。

输入捕捉到的时间标记可用来计算频率、占空比及信号的其它特征，以及为触发事件创建日志。测量外部信号的占空比时要求每次捕捉后都要改变触发沿，因此读取 ICR3 值以后须尽快改变触发的信号边沿。

PWM 输出的自动关闭与重启

当设置 TCCR3C 寄存器的 DOC3x 位为高时，PWM 输出的自动关闭功能会被使能，满足触发条件时，硬件会清零相应的 COM3x 位，将 PWM 输出信号 OC3x 与其输出引脚断开，切换成通用 IO 输出，实现 PWM 输出的自动关闭。此时，输出引脚的状态可由通用 IO 口的输出来控制。

PWM 输出的自动关闭被使能后，还需要设置其触发条件，由 TCCR3D 寄存器的 DSX3n 位来选择触发源。触发源有模拟比较器中断，外部中断，引脚电平变化中断以及定时器溢出中断，具体情形请参考 TCCR3D 寄存器描述。当某个或某些触发源被选用作为触发条件后，在这些中断标志位被置位的同时，硬件会清零 COM3x 位来关闭 PWM 的输出。

当发生了触发事件关闭 PWM 输出后，定时器模块没有相应的中断标志位，软件需要通过读取触发源的中断标志位来得知触发条件和触发事件。

当 PWM 输出被自动关闭而需要再次重启输出时，软件只需要重新设置 COM3x 位，来切换 OC3x 信号输出到相应的引脚上。需要注意的是，发生自动关闭后，定时器并未停止工作，OC3x 信号的状态也一直在更新。软件可在定时器发生溢出或比较匹配后，再设置 COM3x 位来输出 OC3x 信号，这样可以获得明确的 PWM 输出状态。

死区时间控制

设置 DTEN3 位为“1”时，插入死区时间的功能被使能，OC3A 和 OC3B 的输出波形将在 B 通道比较输出所产生的波形基础上插入设定的死区时间，时间的长度为 DTR3 寄存器的计数时钟数所对应的时间值。如下图所示，OC3A 和 OC3B 的死区时间插入均是以通道 B 的比较输出波形为基准。当 COM3A 和 COM3B 同为“2”或“3”时，OC3A 的波形极性与 OC3B 的波形极性相同，当 COM3A 和 COM3B 分别为“2”或“3”时，OC3A 的波形与 OC3B 的波形极性相反。

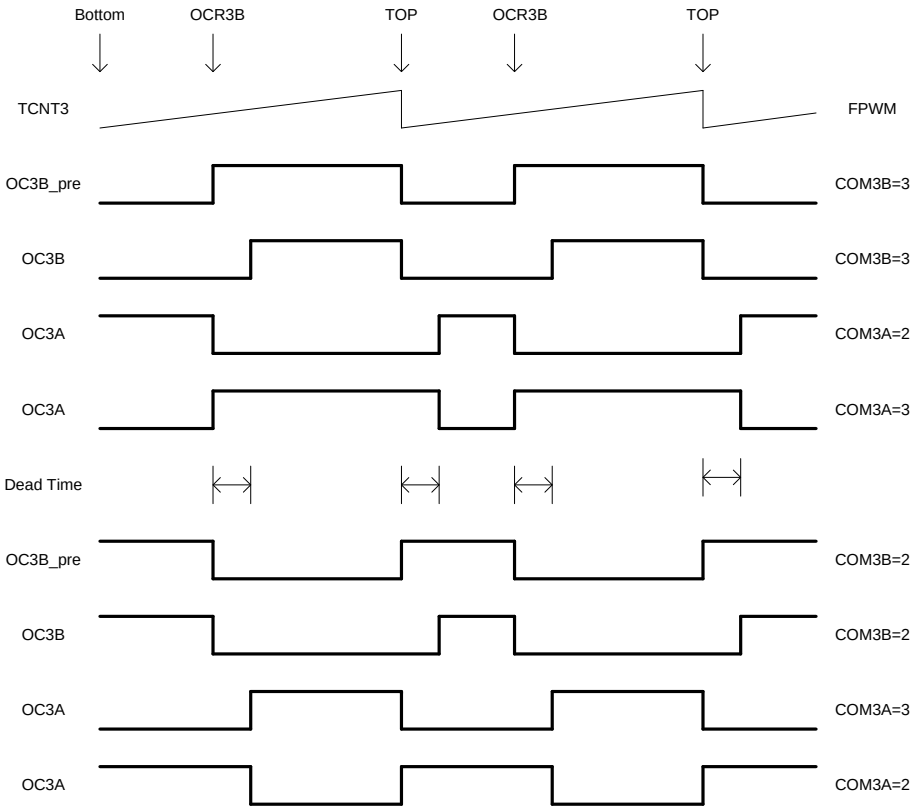


Figure 7 FPWM 模式下 TC3 死区时间控制

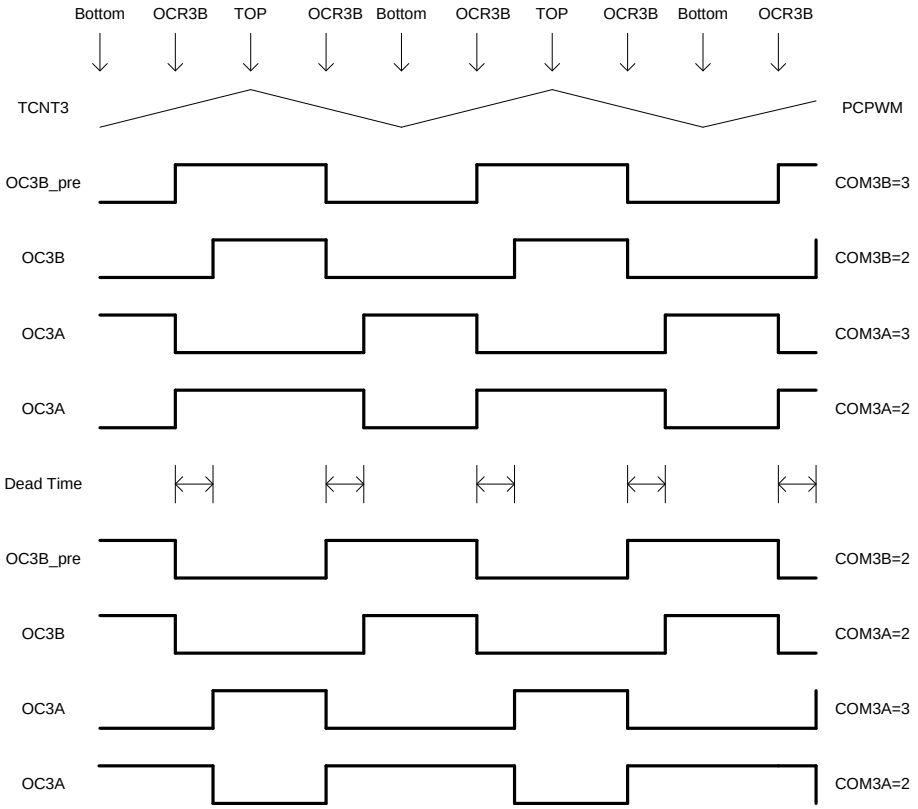


Figure 8 PCPWM 模式下 TC3 死区时间控制

设置 DTEN3 位为“0”时，插入死区时间的功能被禁止，OC3A 和 OC3B 的输出波形为各自比较输出所产生的波形。

寄存器定义

TC3 寄存器列表

寄存器	地址	默认值	描述
TCCR3A	0x90	0x00	TC3 控制寄存器 A
TCCR3B	0x91	0x00	TC3 控制寄存器 B
TCCR3C	0x92	0x00	TC3 控制寄存器 C
TCCR3D	0x93	0x00	TC3 控制寄存器 D
TCNT3L	0x94	0x00	TC3 计数值寄存器低字节
TCNT3H	0x95	0x00	TC3 计数值寄存器高字节
ICR3L	0x96	0x00	TC3 输入捕捉寄存器低字节
ICR3H	0x97	0x00	TC3 输入捕捉寄存器高字节
OCR3AL	0x98	0x00	TC3 输出比较寄存器 A 低字节
OCR3AH	0x99	0x00	TC3 输出比较寄存器 A 高字节
OCR3BL	0x9A	0x00	TC3 输出比较寄存器 B 低字节
OCR3BH	0x9B	0x00	TC3 输出比较寄存器 B 高字节
DTR3L	0x9C	0x00	TC3 死区时间寄存器低字节
DTR3H	0x9D	0x00	TC3 死区时间寄存器高字节
OCR3CL	0x9E	0x00	TC3 输出比较寄存器 C 低字节

OCR3CH	0x9F	0x00	TC3 输出比较寄存器 C 高字节
TIMSK3	0x71	0x00	定时计数器中断屏蔽寄存器
TIFR3	0x38	0x00	定时计数器中断标志寄存器

TCCR3A–TC3 控制寄存器 A

TCCR3A –TC3 控制寄存器 A								
地址: 0x90					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	COM3A1	COM3A0	COM3B1	COM3B0	COM3C1	COM3C0	WGM31	WGM30
R/W	R/W	R/W	R/W	R/W	W	W	R/W	R/W

Bit	Name	描述
7	COM3A1	比较匹配输出 A 模式控制高位。 COM3A1 和 COM3A0 组成 COM3A[1:0]来控制输出比较波形 OC3A。如果 COM3A 的 1 位或者 2 位都置位，输出比较波形占据着 OC3A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM3A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
6	COM3A0	比较匹配输出 A 模式控制低位。 COM3A1 和 COM3A0 组成 COM3A[1:0]来控制输出比较波形 OC3A。如果 COM3A 的 1 位或者 2 位都置位，输出比较波形占据着 OC3A 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM3A 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
5	COM3B1	比较匹配输出 B 模式控制高位。 COM3B1 和 COM3B0 组成 COM3B[1:0]来控制输出比较波形 OC3B。如果 COM3B 的 1 位或者 2 位都置位，输出比较波形占据着 OC3B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM3B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
4	COM3B0	比较匹配输出 B 模式控制低位。 COM3B1 和 COM3B0 组成 COM3B[1:0]来控制输出比较波形 OC3B。如果 COM3B 的 1 位或者 2 位都置位，输出比较波形占据着 OC3B 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM3B 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
3	COM3C1	比较匹配输出 C 模式控制高位。 COM3C1 和 COM3C0 组成 COM3C[1:0]来控制输出比较波形 OC3C。如果 COM3C 的 1 位或者 2 位都置位，输出比较波形占据着 OC3C 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM3C 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
2	COM3C0	比较匹配输出 C 模式控制低位。 COM3C1 和 COM3C0 组成 COM3C[1:0]来控制输出比较波形 OC3C。如果

		COM3C 的 1 位或者 2 位都置位，输出比较波形占据着 OC3C 引脚，不过该引脚的数据方向寄存器必须置高才能输出此波形。在不同工作模式下，COM3C 对输出比较波形的控制也不同，具体见比较输出模式控制表格描述。
1	WGM31	波形产生模式控制次低位。 WGM31 和 WGM33,WGM32,WGM30 一起组成波形产生模式控制 WGM3[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
0	WGM30	波形产生模式控制最低位。 WGM30 和 WGM33,WGM32,WGM31 一起组成波形产生模式控制 WGM3[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。

下表为非 PWM 模式（即普通模式和 CTC 模式）下，比较输出模式对输出比较波形的控制。

非 PWM 模式下比较输出模式控制

COM3x[1:0]	描述
0	OC3x 断开，通用 IO 口操作
1	比较匹配时翻转 OC3x 信号
2	比较匹配时清零 OC3x 信号
3	比较匹配时置位 OC3x 信号

下表为快速 PWM 模式下比较输出模式对输出比较波形的控制。

快速 PWM 模式下比较输出模式控制

COM3x[1:0]	描述
0	OC3x 断开，通用 IO 口操作
1	WGM3 为 15 时：比较匹配时翻转 OC3A 信号，OC3B 断开 WGM3 为其它值时：OC3x 断开，通用 IO 口操作
2	比较匹配时清零 OC3x 信号，最大值匹配时置位 OC3x 信号
3	比较匹配时置位 OC3x 信号，最大值匹配时清零 OC3x 信号

下表为相位修正模式下比较输出模式对输出比较波形的控制。

相位修正和相位频率修正 PWM 模式下比较输出模式控制

COM3x[1:0]	描述
0	OC3x 断开，通用 IO 口操作
1	WGM3 为 9 或 11 时：比较匹配时翻转 OC3A 信号，OC3B 断开 WGM3 为其它值时：OC3x 断开，通用 IO 口操作
2	升序计数下比较匹配清零 OC3x 信号，降序计数下比较匹配置位 OC3x 信号
3	升序计数下比较匹配置位 OC3x 信号，降序计数下比较匹配清零 OC3x 信号

TCCR3B–TC3 控制寄存器 B

TCCR3B–TC3 控制寄存器 B

地址: 0x91					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	ICNC3	ICES3	-	WGM33	WGM32	CS32	CS31	CS30
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	ICNC3	输入捕捉噪声抑制器使能控制位。 当设置 ICNC3 位为“1”时，使能输入捕捉噪声抑制器，此时外部引脚 ICP3 的输入被滤波，连续 4 个采样值相等时输入信号才有效，该功能使得输入捕捉被延迟了 4 个时钟周期。 当设置 ICNC3 位为“0”时，禁止输入捕捉噪声抑制器，此时外部引脚 ICP3 的输入直接有效。
6	ICES3	输入捕捉触发沿选择控制位。 当设置 ICES3 位为“1”时，选择电平的上升沿触发输入捕捉；当设置 ICES3 位为“0”时，选择电平的下降沿触发输入捕捉。 当捕获到一个事件后，计数器的数值被复制到 ICR3 寄存器，同时置位输入捕捉标志 ICF3。如果中断使能，产生输入捕捉中断。
5	-	保留。
4	WGM33	波形产生模式控制高位。 WGM33 和 WGM32,WGM31,WGM30 一起组成波形产生模式控制 WGM3[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
3	WGM32	波形产生模式控制次高位。 WGM32 和 WGM33,WGM31,WGM30 一起组成波形产生模式控制 WGM3[3:0]，控制计数器的计数方式和波形产生方式，具体见波形产生模式表格描述。
2	CS32	时钟选择控制高位。 用于选择定时计数器 3 的时钟源。
1	CS31	时钟选择控制中位。 用于选择定时计数器 3 的时钟源。
0	CS30	时钟选择控制低位。 用于选择定时计数器 3 的时钟源。

CS3[2:0]	描述
0	无时钟源，停止计数
1	clk_{sys}
2	$clk_{sys}/8$，来自预分频器
3	$clk_{sys}/64$，来自预分频器
4	$clk_{sys}/256$，来自预分频器
5	$clk_{sys}/1024$，来自预分频器
6	外部时钟 T3 引脚，下降沿触发
7	外部时钟 T3 引脚，上升沿触发

下表为波形产生模式控制。

Table 5 波形产生模式控制

WGM3[3:0]	工作模式	TOP 值	更新 OCR1A 时刻	置位 TOV3 时刻

0	Normal	0xFFFF	立即	MAX
1	8 位 PCPWM	0x00FF	TOP	BOTTOM
2	9 位 PCPWM	0x01FF	TOP	BOTTOM
3	10 位 PCPWM	0x03FF	TOP	BOTTOM
4	CTC	OCR3A	立即	MAX
5	8 位 FPWM	0x00FF	BOTTOM	TOP
6	9 位 FPWM	0x01FF	BOTTOM	TOP
7	10 位 FPWM	0x03FF	BOTTOM	TOP
8	PFCPWM	ICR3	BOTTOM	BOTTOM
9	PFCPWM	OCR3A	BOTTOM	BOTTOM
10	PCPWM	ICR3	TOP	BOTTOM
11	PCPWM	OCR3A	TOP	BOTTOM
12	CTC	ICR3	立即	MAX
13	保留	-	-	-
14	FPWM	ICR3	TOP	TOP
15	FPWM	OCR3A	TOP	TOP

TCCR3C–TC3 控制寄存器 C

TCCR3C–TC3 控制寄存器 C								
地址: 0x92					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	FOC3A	FOC3B	DOC3B	DOC3A	DTEN3	-	DOC3C	FOC3C
R/W	W	W	-	-	-	-	-	-

Bit	Name	描述
7	FOC3A	强制输出比较 A。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC3A 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF3A 标志，也不会重载或清零定时器，但是输出引脚 OC3A 将被按照 COM3A 的设置相应的更新，就跟真的发生了比较匹配一样。 工作于 PWM 模式时，写 TCCR3A 寄存器时要对其清零。 读取 FOC3A 的返回值一直为零。
6	FOC3B	强制输出比较 B。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC3B 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF3B 标志，也不会重载或清零定时器，但是输出引脚 OC3B 将被按照 COM3B 的设置相应的更新，就跟真的发生了比较匹配一样。 工作于 PWM 模式时，写 TCCR3A 寄存器时要对其清零。 读取 FOC3B 的返回值一直为零。
5	DOC3B	禁止输出比较 B 使能控制位。 当 DOC3B 位为高时，硬件禁止输出比较 B 被使能，在满足禁止输出的条件后，COM3B 位会被清零，输出引脚 OC3B 断开，该引脚变成通用 IO 操作。 当 DOC3B 位为低时，硬件禁止输出比较 B 功能无效。
4	DOC3A	禁止输出比较 A 使能控制位。

		当 DOC3A 位为高时，硬件禁止输出比较 A 被使能，在满足禁止输出的条件后，COM3A 位会被清零，输出引脚 OC3A 断开，该引脚变成通用 IO 操作。 当 DOC3A 位为低时，硬件禁止输出比较 A 功能无效。
3	DTEN3	死区时间使能控制位。 当 DTEN3 位为高时，死区时间被使能，OC3A 和 OC3B 变成互补输出，并按 DTR3L 和 DTR3H 所设定的来插入死区时间。 当 DTEN3 位为低时，死区时间被禁止。OC3A 和 OC3B 均为单路输出。
2	-	
1	DOC3C	禁止输出比较 C 使能控制位。 当 DOC3C 位为高时，硬件禁止输出比较 C 被使能，在满足禁止输出的条件后，COM3C 位会被清零，输出引脚 OC3C 断开，该引脚变成通用 IO 操作。 当 DOC3C 位为低时，硬件禁止输出比较 C 功能无效。
0	FOC3C	强制输出比较 C。 工作于非 PWM 模式时，可以通过对强制输出比较位 FOC3C 写“1”的方式来产生比较匹配。强制比较匹配不会置位 OCF3C 标志，也不会重载或清零定时器，但是输出引脚 OC3C 将被按照 COM3C 的设置相应的更新，就跟真的发生了比较匹配一样。 工作于 PWM 模式时，写 TCCR3A 寄存器时要对其清零。 读取 FOC3C 的返回值一直为零。

TCCR3D–TC3 控制寄存器 D

TCCR3D–TC3 控制寄存器 D								
地址: 0x93					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	DSX37	DSX36	DSX35	DSX34	-	-	DSX31	DSX30
R/W	R/W	R/W	R/W	R/W	-	-	R/W	R/W

Bit	Name	描述
7	DSX37	TC3 触发源选择控制使能第 7 位。 当设置 DSX37 位为“1”时，TC0 溢出作为为关闭输出比较信号波形 OC3x 的触发源被使能。当 DOC3x 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC3x 的波形输出。 当设置 DSX37 位为“0”时，TC0 溢出作为为关闭输出比较信号波形 OC3x 的触发源被禁止。
6	DSX36	TC3 触发源选择控制使能第 6 位。 当设置 DSX36 位为“1”时，TC2 溢出作为为关闭输出比较信号波形 OC3x 的触发源被使能。当 DOC3x 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC3x 的波形输出。 当设置 DSX36 位为“0”时，TC2 溢出作为为关闭输出比较信号波形 OC3x 的触发源被禁止。
5	DSX35	TC3 触发源选择控制使能第 5 位。 当设置 DSX35 位为“1”时，引脚电平变化 1 作为为关闭输出比较信号波形 OC3x 的触发源被使能。当 DOC3x 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC3x 的波形输出。

		当设置 DSX35 位为“0”时，引脚电平变化 1 作为为关闭输出比较信号波形 OC3x 的触发源被禁止。
4	DSX34	TC3 触发源选择控制使能第 4 位。 当设置 DSX34 位为“1”时，外部中断 1 作为为关闭输出比较信号波形 OC3x 的触发源被使能。当 DOC3x 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC3x 的波形输出。 当设置 DSX34 位为“0”时，外部中断 1 作为为关闭输出比较信号波形 OC3x 的触发源被禁止。
3:2	-	保留。
1	DSX31	TC3 触发源选择控制使能第 1 位。 当设置 DSX31 位为“1”时，模拟比较器 1 作为为关闭输出比较信号波形 OC3x 的触发源被使能。当 DOC3x 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC3x 的波形输出。 当设置 DSX31 位为“0”时，模拟比较器 1 作为为关闭输出比较信号波形 OC3x 的触发源被禁止。
0	DSX30	TC3 触发源选择控制使能第 0 位。 当设置 DSX30 位为“1”时，模拟比较器 0 作为为关闭输出比较信号波形 OC3x 的触发源被使能。当 DOC3x 位为“1”时，所选触发源的中断标志寄存器位的上升沿就会自动关闭 OC3x 的波形输出。 当设置 DSX30 位为“0”时，模拟比较器 0 作为为关闭输出比较信号波形 OC3x 的触发源被禁止。

下表为波形输出的触发源的选择控制。

关闭 OC3x 波形输出的触发源选择控制

DOC3x	DSX3n=1	触发源	描述
0	-	-	DOC3x 位为“0”，触发源关闭波形输出功能被禁止
1	0	模拟比较器 0	ACIF0 的上升沿将关闭 OC3x 波形输出
1	1	模拟比较器 1	ACIF1 的上升沿将关闭 OC3x 波形输出
1	4	外部中断 1	INTF1 的上升沿将关闭 OC3x 波形输出
1	5	引脚电平变化 1	PCIF1 的上升沿将关闭 OC3x 波形输出
1	6	TC2 溢出	TOV2 的上升沿将关闭 OC3x 波形输出
1	7	TC0 溢出	TOV0 的上升沿将关闭 OC3x 波形输出

注意：

2） DSX3n=1 表示 TCCR1D 寄存器的第 n 位为 1 时，各寄存器位可同时置位。

TCNT3L–TC3 计数器寄存器低字节

TCNT3L –TC3 计数值寄存器低字节								
地址: 0x94					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	TCNT3L7	TCNT3L6	TCNT3L5	TCNT3L4	TCNT3L3	TCNT3L2	TCNT3L1	TCNT3L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	TCNT3L	TC3 计数值的低字节。						

		TCNT3H 和 TCNT3L 结合到一起组成 TCNT3，通过 TCNT3 寄存器可以直接对计数器的 16 位计数值进行读写访问。读写 16 位寄存器需要两次操作。写 16 位 TCNT3 时，应先写入 TCNT3H。读 16 位 TCNT3 时，应先读取 TCNT3L。 CPU 对 TCNT3 寄存器的写操作会在下一个定时器时钟周期阻止比较匹配的发生，即使定时器已经停止。这就允许初始化 TCNT3 寄存器的值与 OCR3x 的值一致而不会引发中断。 如果写入 TCNT3 的数值等于或绕过 OCR3x 值时，比较匹配就会丢失，造成不正确的波形发生结果。 没有选择时钟源时定时器停止计数，但 CPU 仍可以访问 TCNT3。CPU 写计数器比清零或加减操作的优先级高。

TCNT3H–TC3 计数器寄存器高字节

TCNT3H –TC3 计数值寄存器高字节								
地址: 0x95					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	TCNT3H7	TCNT3H6	TCNT3H5	TCNT3H4	TCNT3H3	TCNT3H2	TCNT3H1	TCNT3H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	TCNT3H	TC3 计数值的高字节。 TCNT3H 和 TCNT3L 结合到一起组成 TCNT3，通过 TCNT3 寄存器可以直接对计数器的 16 位计数值进行读写访问。读写 16 位寄存器需要两次操作。写 16 位 TCNT3 时，应先写入 TCNT3H。读 16 位 TCNT3 时，应先读取 TCNT3L。 CPU 对 TCNT3 寄存器的写操作会在下一个定时器时钟周期阻止比较匹配的发生，即使定时器已经停止。这就允许初始化 TCNT3 寄存器的值与 OCR3x 的值一致而不会引发中断。 如果写入 TCNT3 的数值等于或绕过 OCR3x 值时，比较匹配就会丢失，造成不正确的波形发生结果。 没有选择时钟源时定时器停止计数，但 CPU 仍可以访问 TCNT3。CPU 写计数器比清零或加减操作的优先级高。

ICR3L–TC3 俘获寄存器低字节

ICR3L –TC3 输入捕捉寄存器低字节								
地址: 0x96					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ICR3L7	ICR3L6	ICR3L5	ICR3L4	ICR3L3	ICR3L2	ICR3L1	ICR3L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	ICR3L	TC3 输入捕捉值的低字节。 ICR3H 和 ICR3L 结合到一起组成 16 位的 ICR3。读写 16 位寄存器需要两次操作。写 16 位 ICR3 时，应先写入 ICR3H。读 16 位 ICR3 时，应先读取 ICR3L。 当输入捕捉被触发时，计数值 TCNT3 就会更新复制到 ICR3 寄存器里。ICR3 寄存器也可用来定义计数的 TOP 值。

ICR3H–TC3 俘获寄存器高字节

ICR3H –TC3 输入捕捉寄存器高字节								
地址: 0x97					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ICR3H7	ICR3H6	ICR3H5	ICR3H4	ICR3H3	ICR3H2	ICR3H1	ICR3H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	ICR3H	TC3 输入捕捉值的高字节。 ICR3H 和 ICR3L 结合到一起组成 16 位的 ICR3。读写 16 位寄存器需要两次操作。写 16 位 ICR3 时，应先写入 ICR3H。读 16 位 ICR3 时，应先读取 ICR3L。 当输入捕捉被触发时，计数值 TCNT3 就会更新复制到 ICR3 寄存器里。ICR3 寄存器也可用来定义计数的 TOP 值。						

OCR3AL–TC3 输出比较寄存器 A 低字节

OCR3AL –TC3 输出比较寄存器 A 低字节								
地址: 0x98					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3AL7	OCR3AL6	OCR3AL5	OCR3AL4	OCR3AL3	OCR3AL2	OCR3AL1	OCR3AL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR3AL	输出比较寄存器 A 的低字节。 OCR3AL 和 OCR3AH 结合到一起组成 16 位的 OCR3A。读写 16 位寄存器需要两次操作。写 16 位 OCR3A 时，应先写入 OCR3AH。读 16 位 OCR3A 时，应先读取 OCR3AL。 OCR3A 不间断地与计数器数值 TCNT3 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC3A 引脚上产生波形。 当使用 PWM 模式时，OCR3A 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR3A 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR3A 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR3A 本身。						

OCR3AH–TC3 输出比较寄存器 A 高字节

OCR3AH –TC3 输出比较寄存器 A 高字节								
地址: 0x99					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	OCR3AH7	OCR3AH6	OCR3AH5	OCR3AH4	OCR3AH3	OCR3AH2	OCR3AH1	OCR3AH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR3AH	输出比较寄存器 A 的高字节。 OCR3AL 和 OCR3AH 结合到一起组成 16 位的 OCR3A。读写 16 位寄存器需要两次操作。写 16 位 OCR3A 时，应先写入 OCR3AH。读 16 位 OCR3A 时，应先读取 OCR3AL。 OCR3A 不间断地与计数器数值 TCNT3 进行比较。比较匹配可以用来产生输出比较中断，或						

		者用来在 OC3A 引脚上产生波形。 当使用 PWM 模式时，OCR3A 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR3A 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR3A 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR3A 本身。

OCR3BL–TC3 输出比较寄存器 B 低字节

OCR3BL –TC3 输出比较寄存器 B 低字节								
地址: 0x9A					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3BL7	OCR3BL6	OCR3BL5	OCR3BL4	OCR3BL3	OCR3BL2	OCR3BL1	OCR3BL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR3BL	输出比较寄存器 B 的低字节。 OCR3BL 和 OCR3BH 结合到一起组成 16 位的 OCR3B。读写 16 位寄存器需要两次操作。写 16 位 OCR3B 时，应先写入 OCR3BH。读 16 位 OCR3B 时，应先读取 OCR3BL。 OCR3B 不间断地与计数器数值 TCNT3 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC3B 引脚上产生波形。 当使用 PWM 模式时，OCR3B 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR3B 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR3B 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR3B 本身。							

OCR3BH–TC3 输出比较寄存器 B 高字节

OCR3BH –TC3 输出比较寄存器 B 高字节								
地址: 0x9B					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3BH7	OCR3BH6	OCR3BH5	OCR3BH4	OCR3BH3	OCR3BH2	OCR3BH1	OCR3BH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR3BH	输出比较寄存器 B 的高字节。 OCR3BL 和 OCR3BH 结合到一起组成 16 位的 OCR3B。读写 16 位寄存器需要两次操作。写 16 位 OCR3B 时，应先写入 OCR3BH。读 16 位 OCR3B 时，应先读取 OCR3BL。 OCR3B 不间断地与计数器数值 TCNT3 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC3B 引脚上产生波形。 当使用 PWM 模式时，OCR3B 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR3B 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR3B 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR3B 本身。							

OCR3CL–TC3 输出比较寄存器 C 低字节

OCR3CL –TC3 输出比较寄存器 C 低字节								
地址: 0x9E					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3CL7	OCR3CL6	OCR3CL5	OCR3CL4	OCR3CL3	OCR3CL2	OCR3CL1	OCR3CL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR3CL	输出比较寄存器 C 的低字节。 OCR3CL 和 OCR3CH 结合到一起组成 16 位的 OCR3C。读写 16 位寄存器需要两次操作。写 16 位 OCR3C 时，应先写入 OCR3CH。读 16 位 OCR3C 时，应先读取 OCR3CL。 OCR3C 不间断地与计数器数值 TCNT3 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC3C 引脚上产生波形。 当使用 PWM 模式时，OCR3C 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR3C 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR3C 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR3C 本身。							

OCR3CH–TC3 输出比较寄存器 C 高字节

OCR3CH –TC3 输出比较寄存器 C 高字节								
地址: 0x9F					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	OCR3CH7	OCR3CH6	OCR3CH5	OCR3CH4	OCR3CH3	OCR3CH2	OCR3CH1	OCR3CH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	OCR3CH	输出比较寄存器 C 的高字节。 OCR3CL 和 OCR3CH 结合到一起组成 16 位的 OCR3C。读写 16 位寄存器需要两次操作。写 16 位 OCR3C 时，应先写入 OCR3CH。读 16 位 OCR3C 时，应先读取 OCR3CL。 OCR3C 不间断地与计数器数值 TCNT3 进行比较。比较匹配可以用来产生输出比较中断，或者用来在 OC3C 引脚上产生波形。 当使用 PWM 模式时，OCR3C 寄存器使用双缓冲寄存器。而普通工作模式和匹配清零模式下，双缓冲功能是禁止的。双缓冲可以将更新 OCR3C 寄存器与计数最大值或最小值时刻同步起来，从而防止产生不对称的 PWM 脉冲，消除了干扰脉冲。 使用双缓冲功能时，CPU 访问的是 OCR3C 缓冲寄存器，禁止双缓冲功能时 CPU 访问的是 OCR3C 本身。							

DTR3L–TC3 死区时间寄存器低字节

DTR3L –TC3 死区时间寄存器低字节								
地址: 0x9C					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	DTR3L7	DTR3L6	DTR3L5	DTR3L4	DTR3L3	DTR3L2	DTR3L1	DTR3L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	DTR3L	死区时间寄存器低字节。 当 DTEN3 位为高时，OC3A 和 OC3B 为互补输出，OC3A 输出上所插入的死区时间由 DTR3L 个计数时钟决定。

DTR3H–TC3 死区时间寄存器高字节

DTR3H –TC3 死区时间寄存器高字节								
地址: 0x9D					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	DTR3H7	DTR3H6	DTR3H5	DTR3H4	DTR3H3	DTR3H2	DTR3H1	DTR3H0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	DTR3H	死区时间寄存器高字节。 当 DTEN3 位为高时，OC3A 和 OC3B 为互补输出，OC3B 输出上所插入的死区时间由 DTR3H 个计数时钟决定。						

TIMSK3–TC3 中断屏蔽寄存器

TIMSK3 – TC3 中断屏蔽寄存器								
地址: 0x71					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	ICIE3	-	OCIE3C	OCIE3B	OCIE3A	TOIE3
R/W	-	-	R/W	-	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:6	-	保留。						
5	ICIE3	TC3 输入捕捉中断使能控制位。 当 ICIE3 位为“1”时，且全局中断置位，TC3 输入捕捉中断被使能。当输入捕捉触发时，即 TIFR3 的 ICF3 标志被置位，中断发生。 当 ICIE3 位为“0”时，TC3 输入捕捉中断被禁止。						
4	-	保留。						
3	OCIE3C	TC3 输出比较 C 匹配中断使能位。 当 OCIE3C 位为“1”，且全局中断置位，TC3 输出比较 C 匹配中断使能。当比较匹配发生时，即 TIFR3 中 OCF3C 位被置位时，中断产生。 当 OCIE3C 位为“0”时，TC3 输出比较 C 匹配中断被禁止。						
2	OCIE3B	TC3 输出比较 B 匹配中断使能位。 当 OCIE3B 位为“1”，且全局中断置位，TC3 输出比较 B 匹配中断使能。当比较匹配发生时，即 TIFR3 中 OCF3B 位被置位时，中断产生。 当 OCIE3B 位为“0”时，TC3 输出比较 B 匹配中断被禁止。						
1	OCIE3A	TC3 输出比较 A 匹配中断使能位。 当 OCIE3A 位为“1”，且全局中断置位，TC3 输出比较 A 匹配中断使能。当比较匹配发生时，即 TIFR3 中 OCF3A 位被置位时，中断产生。当 OCIE3A 位为“0”时，TC3 输出比较 A 匹配中断被禁止。						
0	TOIE3	TC3 溢出中断使能位。						

		当 TOIE3 位为“1”，且全局中断置位，TC3 溢出中断使能。当 TC3 发生溢出，即 TIFR3 中的 TOV3 位被置位时，中断产生。当 TOIE3 位为“0”时，TC3 溢出中断被禁止。

TIFR3–TC3 中断标志寄存器

TIFR3 – TC3 中断标志寄存器								
地址: 0x38					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	ICF3	-	-	OCF3B	OCF3A	TOV3
R/W	-	-	R/W	-	-	R/W	R/W	R/W

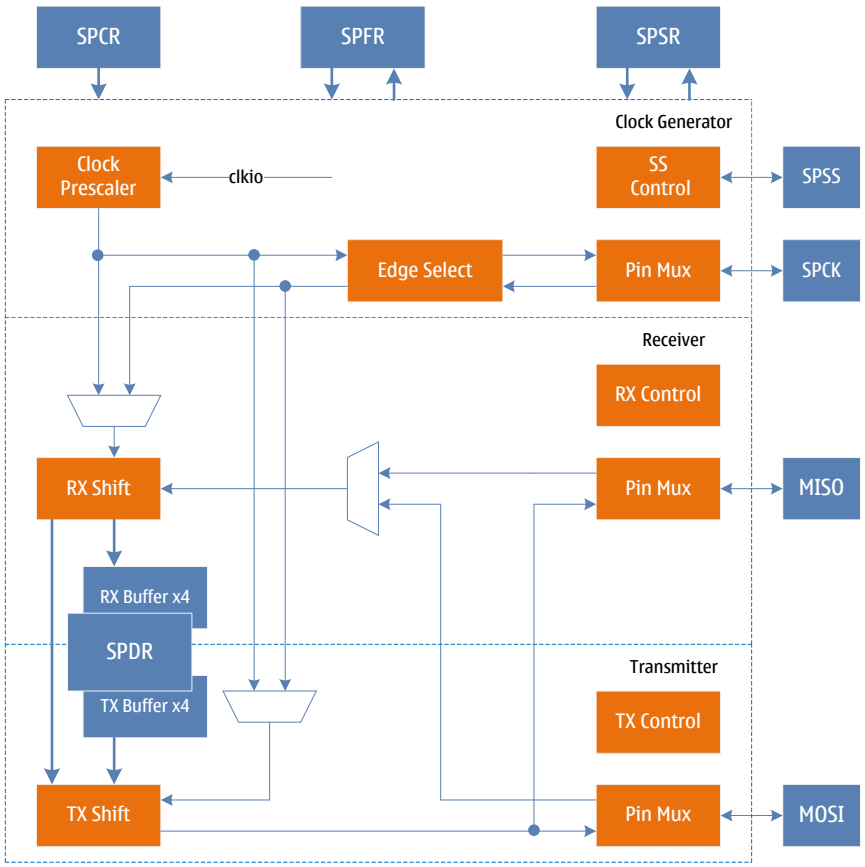
Bit	Name	描述
7:6	-	保留。
5	ICF3	输入捕捉标志位。 当输入捕捉事件发生时，ICF3 标志被置位。当 ICR3 被用作计数的 TOP 值，且计数值到达 TOP 值时，ICF3 标志被置位。若 ICIE1 为“1”且全局中断标志置位，则会产生输入捕捉中断。ICF3 标志位不会自动清零，需要软件对 ICF3 位写“1”清零。
4	-	保留。
3	OCF3C	输出比较 C 匹配标志位。 当 TCNT3 等于 OCR3C 时，比较单元就给出匹配信号，并置位比较标志 OCF3C。若此时输出比较中断使能 OCIE3C 为“1”且全局中断标志置位，则会产生输出比较中断。OCF3 标志位不会自动清零，需要软件 OCF3C 位写“1”清零该位。
2	OCF3B	输出比较 B 匹配标志位。 当 TCNT3 等于 OCR3B 时，比较单元就给出匹配信号，并置位比较标志 OCF3B。若此时输出比较中断使能 OCIE3B 为“1”且全局中断标志置位，则会产生输出比较中断。OCF3B 标志位不会自动清零，需要软件对 OCF3B 位写“1”清零。
1	OCF3A	输出比较 A 匹配标志位。 当 TCNT3 等于 OCR3A 时，比较单元就给出匹配信号，并置位比较标志 OCF3A。若此时输出比较中断使能 OCIE3A 为“1”且全局中断标志置位，则会产生输出比较中断。OCF3A 标志位不会自动清零，需要软件对 OCF3A 位写“1”清零。
0	TOV3	溢出标志位。 当计数器发生溢出时，置位溢出标志 TOV3。若此时溢出中断使能 TOIE3 为“1”且全局中断标志置位，则会产生溢出中断。 TOV3 标志位不会被自动清零，需要软件对 TOV3 位写“1”清零。

同步串行外设接口(SPI)

- 全双工，三线同步数据传输
- 主机或从机操作
- 最低位或最高位优先传输
- 7 种可编程的比特率
- 发送结束中断标志
- 写入冲突标志保护机制
- 可从闲置模式唤醒
- 主机操作时具有倍速模式
- 支持主机双线输入模式
- 输入/输出均有 4 个缓存寄存器

综述

SPI 主要包括三个部分：时钟预分频器，时钟检测器，从机选择检测器，发送器和接收器。



SPI 结构图

控制和状态寄存器由这三个部分共享。时钟预分频器只工作在主机操作模式下，由比特率控制位来选择分频系数，从而产生相应的分频时钟，输出到 SPCK 引脚上。时钟检测器只工作在从机操作模式下，检测从 SPCK 引脚上输入的时钟沿，根据 SPI 的数据传输模式对发送和接收移位寄存器进行移位操作。从机选择检测器对从机选择信号 SPSS 进行检测，得到

传输的状态来控制发送器和接收器的操作。发送器由一个移位寄存器和发送控制逻辑组成。接收器由一个移位寄存器，四个接收缓冲器和接收控制逻辑组成。

时钟产生

时钟产生逻辑分为主机时钟预分频器和从机时钟检测器，分别工作在主机操作和从机操作模式下。时钟预分频器由比特率控制位和倍速控制位来选择分频系数，产生相应的分频时钟（共有 7 种可选的分频系数，详细信息见寄存器描述），输出到 SPCK 引脚为通信提供时钟，同时为内部发送和接收移位寄存器提供移位时钟。时钟检测器对输入时钟 SPCK 进行边沿检测，根据 SPI 的数据传输模式对发送器和接收器进行移位操作。为保证对时钟信号的正确采样，SPCK 时钟的高电平和低电平的宽度均须大于 2 个系统时钟周期。

发送和接收

SPI 模块在单线模式下支持同时发送和接收，在双线模式下只支持主机双线接收。

单线发送和接收

SPI 的主机将需要通信的从机选择信号 SPSS 拉低，即可启动一次传输过程。主机和从机将需要传输的数据准备好，主机在时钟信号 SPCK 上产生时钟脉冲以交换数据，主机的数据从 MOSI 移出，从 MISO 移入，从机的数据从 MISO 移出，从 MOSI 移入，交换完数据后主机拉高 SPSS 信号即可完成通信。

当配置为主机时，SPI 模块并不控制 SPSS 引脚，必须由用户软件来处理。软件拉低 SPSS 引脚，选择要通信的从机，启动传输。软件将需要传输的数据写入 SPDR 寄存器即会启动时钟发生器，硬件产生通信的时钟，并把 8 位数据移出给从机，同时把从机的数据移入。移位一个字节的数据后，停止时钟发生器，并置位传输完成标志 SPIF。软件可再次写入数据到 SPDR 寄存器来继续传输下一个字节，也可以拉高 SPSS 信号来结束当前传输。最后进来的数据将保存在接收缓冲器中。

当配置为从机时，只要 SPSS 信号一直为高，SPI 模块将保持睡眠状态，并保持 MISO 引脚为三态。这时软件可更新 SPDR 寄存器的内容。即使此时 SPCK 引脚上有输入时钟脉冲，SPDR 的数据也不会被移出，直至 SPSS 信号被拉低。当一个字节的数据传输完成之后，硬件置位传输完成标志 SPIF。此时软件在读取移入的数据之前可继续往 SPDR 寄存器写入数据，最后进来的数据将保存在接收缓冲器中。

SPI 模块在发送方向只有四个缓冲器，在接收方向也有四个缓冲器。在发送数据时，当发送缓冲器处于非满状态（即发送缓冲器满标志位 WRFULL 位为低）时，可对 SPDR 寄存器进行写操作。而在接收数据时，当接收缓冲器属于非空状态（即接收缓冲器空标志位 RDEMPT 位为低）时，可通过访问 SPDR 寄存器读取已经接收到的字符。

主机双线接收

SPI 模块的双线模式只在主机操作模式下有效，与单线模式的不同在于 MOSI 和 MISO 都用于主机接收数据，每一个 SPCK 时钟脉冲同时接收 2 个比特的数据（MISO 线上的数据在

前，MOSI 线上的数据在后），接收完两个字节的数据之后硬件置位传输完成标志 SPIF，数据保存到接收缓冲器和移位寄存器中。此时软件须读取 SPDR 寄存器两次来得到所接收的两个字节的数据。需要注意的是，虽然双线模式下主机不向从机发送数据，软件仍需要往 SPDR 寄存器写入数据来启动时钟发生器产生通信时钟，写入一次 SPDR 寄存器即可接收两个字节的数据。

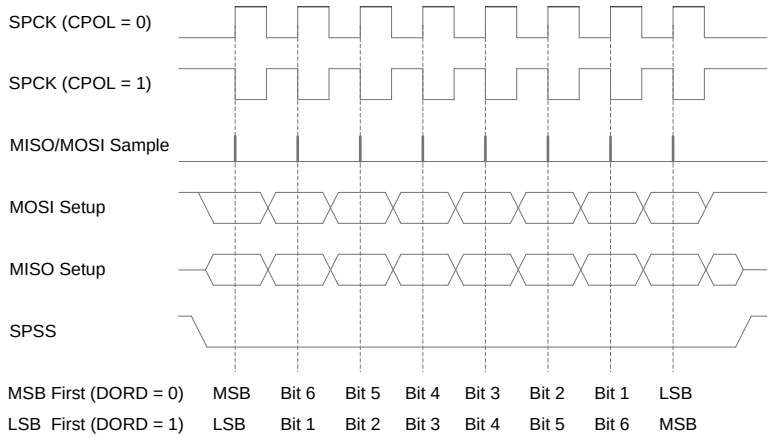
数据模式

单线模式下，相对于串行数据，SPI 有 4 种 SPCK 相位和极性的组合方式，由 CPHA 和 CPOL 来控制，如下表所示。

CPHA 和 CPOL 选择数据传输模式

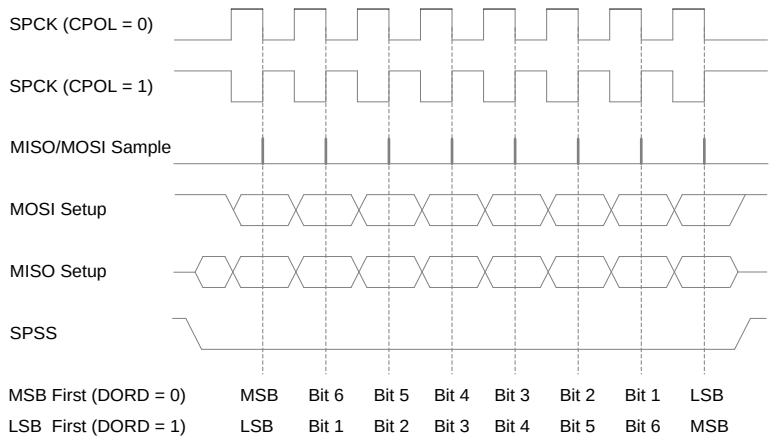
CPOL	CPHA	起始沿	结束沿	SPI 模式
0	0	采样（上升沿）	设置（下降沿）	0
0	1	设置（上升沿）	采样（下降沿）	1
1	0	采样（下降沿）	设置（上升沿）	2
1	1	设置（下降沿）	采样（上升沿）	3

当 CPHA = 0 时，数据采样和设置的时钟沿如下图所示：



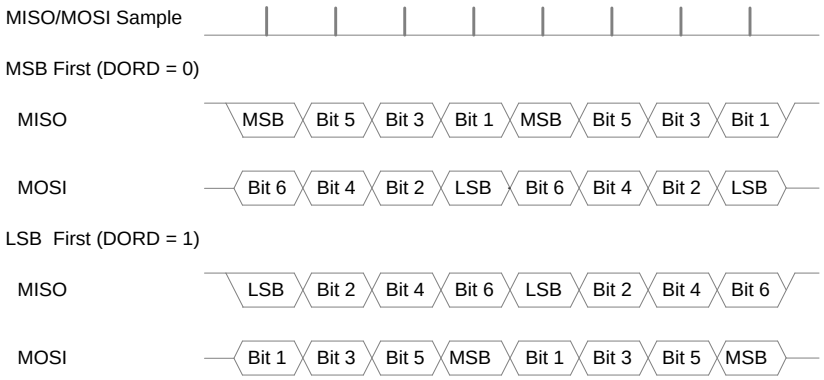
CPHA 为“0”时 SPI 数据传输模式

当 CPHA = 1 时，数据采样和设置的时钟沿如下图所示：



CPHA 为“1”时 SPI 数据传输模式

双线模式下，MISO 和 MISO 均用做主机的输入，数据采样的时刻仍由数据传输模式决定，采样的方式如下图所示：



主机模式下 DUAL 为“1”时 SPI 数据采样模式

SPSS 引脚功能

当配置为从机时，从机选择信号 SPSS 引脚总是作为输入。当 SPSS 引脚保持为低时，SPI 接口被激活，MISO 引脚成为输出引脚（软件进行相应的端口配置），其它引脚均为输入。当 SPSS 引脚保持为高时，SPI 模块被复位，且不再接收数据。SPSS 引脚对于数据包/字节的同步非常有用，可以使从机的位计数器和主机的时钟发生器同步。当 SPSS 拉高时，SPI 从机立即复位接收和发送逻辑，并丢弃移位寄存器里不完整的数据。

当配置为主机时，用户软件可以决定 SPSS 引脚的方向。
若 SPSS 配置为输出，则它可以用来驱动从机的 SPSS 引脚。若 SPSS 配置为输入，必须保持为高以保证主机的正常工作。当配置为主机且 SPSS 引脚为输入，外部电路拉低 SPSS 引脚时，SPI 模块会认为是另外一个主机选择自己作为从机并开始传输数据。为了防止总线冲突，SPI 模块将进行如下动作：

1. 清零位于 SPCR 寄存器的 MSTR 位，转换为从机，从而 MOSI 和 SPCK 变为输入；
2. 置位位于 SPSR 寄存器的 SPIF 位，若中断使能则产生 SPI 中断。

因此，使用中断方式处理 SPI 主机的数据传输，并且存在 SPSS 被拉低的可能性时，中断服务程序应该检查 MSTR 位是否为“1”。若被清零，软件须将其置位，以重新使能 SPI 主机模式。

SPI 初始化

进行通信之前首先要对 SPI 进行初始化。初始化过程通常包括主机从机操作的选择，数据传输模式的设定，比特率的选择，以及各个引脚的方向控制等。其中主机和从机操作下引脚方向的控制各不相同，如下表所示：

引脚方向控制

引脚	主机模式下的方向	从机模式下的方向
MOSI	用户软件定义	输入
MISO	输入	用户软件定义
SPCK	用户软件定义	输入
SPSS	用户软件定义	输入

SPI 主机初始化

SPI 主机模式的初始化过程如下：

1. 置位 MSTR 位，设置比特率选择控制位，数据传输模式，数据传输次序，中断使能与否，以及双线使能与否；
2. 设置 MOSI 和 SPCK 引脚为输出；
3. 置位 SPE 位。

主机模式下，当不希望 SPI 模块被别的主机选择作为从机使用时，可设置 SPSS 引脚为输出。

SPI 从机初始化

SPI 从机模式初始化过程如下：

1. 清零 MSTR 位，设置数据传输模式，数据传输次序，中断使能与否；
2. 设置 MISO 引脚为输出；
3. 置位 SPE 位。

SPI 中断

当发生下列事件之一或多个时，SPI 的中断标志位 SPIF 将会被置位：

1. 当配置为主机且 SPSS 引脚为输入，外部电路拉低 SPSS 引脚；
2. 当发送缓冲器状态为满，软件继续往 SPDR 寄存器写入数据；
3. 当接收缓冲器状态为满；
4. 当写入发送缓冲器中的数据均已发送出去，发送缓冲器状态为空。

当 SPIF 位被置位，且 SPI 中断使能位 SPIE 和全局中断使能位都为高时，会产生 SPI 中断。进入中断服务程序后，硬件会对 SPIF 进行清零。若 SPIF 位是由上述事件中的 1 和 2 来置位的，SPIF 会被清零；若 SPIF 位是由上述事件中的 3 和 4 来置位的，SPIF 并不会被清零，因为接收或发送缓冲器状态未发生改变时，仍会置位 SPIF 位，此时需要通过软件操作来清零。

SPI 中断服务程序中，软件清零 SPIF 位的操作顺序如下：

1）读取 SPIF 位的状态，若为低，表明 SPIF 位已被硬件清零，无需软件再次清零；若为高，继续一下操作；

2）读取 SPFR 寄存器，若 RDFULL 位为高，表明当前接收缓冲器状态为满，读取 SPDR 寄存器获得接收数据，RDFULL 位会变为低，软件可继续读取 SPDR 寄存器获得接收数据，直到 RDEMPT 位为高；

3）读取 SPFR 寄存器，若 RDFULL 位为低，而 WREMPT 位为高，表明当前接收缓冲器状态为非满，而发送缓冲器状态为空，软件可读取 SPDR 寄存器获得接收数据，直到 RDEMPT 位为高；

4）软件获取所接收到的数据后，再执行清零 SPIF 位。因 SPIF 位为只读位，不能直接对 SPIF 位进行清零，而需要先读取 SPSR 寄存器，再访问 SPDR（读或写 SPDR 寄存器）的方式来清零 SPIF 位。

寄存器定义

SPI 寄存器列表

寄存器	地址	默认值	描述
SPCR	0x4C	0x00	SPI 控制寄存器
SPSR	0x4D	0x00	SPI 状态寄存器
SPDR	0x4E	0x00	SPI 数据寄存器
SDFR	0x39	0x00	SPI 缓冲寄存器

SPCR – SPI 控制寄存器

SPCR – SPI 控制寄存器								
地址: 0x4C					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	SPIE	SPI 中断使能位。 当设置 SPIE 位为“1”时，SPI 中断被使能。当位于 SPSR 寄存器中的 SPIF 位被置位且全局中断使能时，产生 SPI 中断。 当设置 SPIE 位为“0”时，SPI 中断被禁止。
6	SPE	SPI 使能位。 当设置 SPE 位为“1”时，SPI 模块被使能。进行任何 SPI 操作之前必须置位 SPE。 当设置 SPE 位为“0”时，SPI 模块被禁止。
5	DORD	数据次序控制位。 当设置 DORD 位为“1”时，数据的 LSB 首先发送。 当设置 DORD 位为“0”时，数据的 MSB 首先发送。
4	MSTR	主机从机选择控制位。 当设置 MSTR 位为“1”时，选择为主机模式。 当设置 MSTR 位为“0”时，选择为从机模式。 主机模式下，SPSS 引脚配置为输入且被拉低时，MSTR 位将被清零，位于 SPSR 寄存器的 SPIF 被置位，用户必须重新设置 MSTR 进入主机模式。
3	CPOL	时钟极性控制位。 当设置 CPOL 位为“1”时，空闲状态下 SPCK 为高电平。 当设置 CPOL 位为“0”时，空闲状态下 SPCK 为低电平。

CPOL	起始沿	结束沿
0	上升沿	下降沿
1	下降沿	上升沿

Bit	Name	描述
2	CPHA	时钟相位控制位。 当设置 CPHA 位为“1”时，起始沿设置数据，结束沿采样数据。 当设置 CPHA 位为“0”时，起始沿采样数据，结束沿设置数据。

CPHA	起始沿	结束沿

		0	采样	设置
		1	设置	采样
1	SPR1	时钟速率选择位 1。 SPR1 和 SPR0 用来选择 SPI 传输的时钟速率。具体控制方式见 SPCK 和系统时钟的关系表格。		
0	SPR0	时钟速率选择位 0。 SPR1 和 SPR0 用来选择 SPI 传输的时钟速率。具体控制方式见 SPCK 和系统时钟的关系表格。		

SPSR – SPI 状态寄存器

SPSR – SPI 状态寄存器								
地址: 0x4D					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	SPIF	WCOL	-	-	-	DUAL	-	SPI2X
R/W	R	R	R	R	R	R/W	R	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	描述
7	SPIF	SPI 中断标志位。 串行传输结束后置位 SPIF 标志，主机模式下，配置 SPSS 引脚为输入且被拉低时，SPIF 也将被置位。若此时 SPCR 寄存器的 SPIE 位和全局中断使能位都被置位，SPI 中断产生。进入中断服务程序后 SPIF 位自动清零，或者通过先读取 SPSR 寄存器再访问 SPDR 寄存器来清零 SPIF 位。
6	WCOL	写冲突标志位。 在数据传输的过程中写 SPDR 寄存器将置位 WCOL 位。WCOL 位可以通过先读取 SPSR 寄存器再访问 SPDR 寄存器来清零。
5	-	保留。
4	-	保留。
3	-	保留。
2	DUAL	双线模式控制位。 当设置 DUAL 位为“1”时，使能 SPI 双线传输模式。 当设置 DUAL 位为“0”时，禁止 SPI 双线传输模式。 双线传输模式只在 SPI 主机模式下有效，MISO 和 MOSI 均用作主机数据输入，数据的传输方式见主机双线接收和数据模式章节描述。
1	-	保留。
0	SPI2X	SPI 倍速控制位。 当设置 SPI2X 位为“1”时，SPI 的传输速度加倍。 当设置 SPI2X 位为“0”时，SPI 的传输速度不加倍。 具体控制方式见 SPCK 和系统时钟的关系表格。

下表为 SPCK 和系统时钟的关系。

SPCK 和系统时钟的关系

SPI2X	SPR1	SPR0	SPCK 的频率
0	0	0	$f_{sys}/4$
0	0	1	$f_{sys}/16$
0	1	0	$f_{sys}/64$
0	1	1	$f_{sys}/128$
1	0	0	$f_{sys}/2$
1	0	1	$f_{sys}/8$
1	1	0	$f_{sys}/32$
1	1	1	$f_{sys}/64$

SPDR – SPI 数据寄存器

SPDR – SPI 数据寄存器								
地址: 0x4E					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	SPDR7	SPDR6	SPDR5	SPDR4	SPDR3	SPDR2	SPDR1	SPDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	SPDR	SPI 发送和接收的数据。 SPI 发送数据和接收数据共享 SPI 数据寄存器 SPDR。将数据写入 SPDR 即写入发送数据移位寄存器器，从 SPDR 读取数据即读取接收数据缓冲器。

SPFR – SPI 缓冲寄存器

SPFR – SPI 缓冲寄存器								
地址: 0x39					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	RDFULl	RDEMPT	RDPTR1	RDPTR0	WRFULL	WREMPT	WRPTR1	WRPTR0
R/W	R	R/W	R	R	R	R/W	R	R

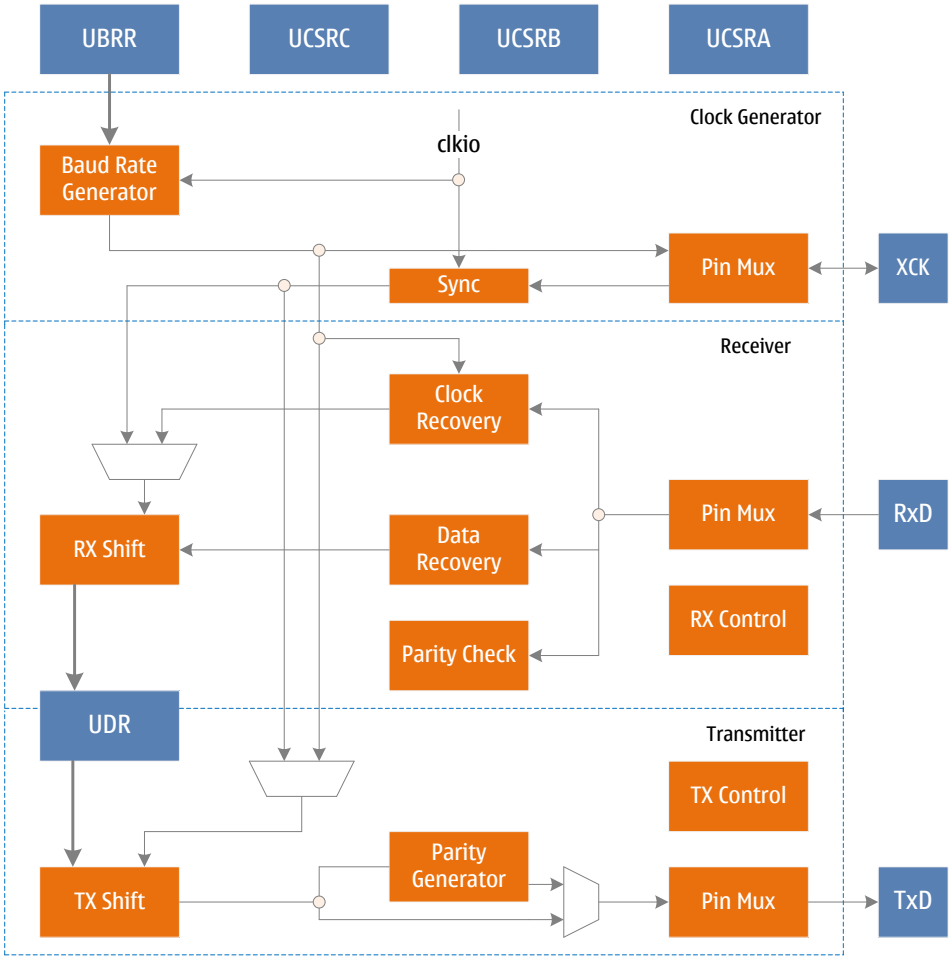
Bit	Name	描述
7	RDFULL	接收缓冲器满标志位。 当接收缓冲器中的数据达到四个字节时，RDFULL 位为高，表明接收缓冲器为满，同时会置位中断标志位。若软件未及时读走接收缓冲器中的数据，再次接收到数据时，接收缓冲器发生溢出，之前的数据会被新数据覆盖。 当接收缓冲器中的数据少于四个字节时，RDFULL 位为低，表明接收缓冲器为非满，还可以接收数据。 当同时对 RDEMPT 位和 WREMPT 位进行置位操作时，接收和发送缓冲器地址以及 SPI 移位寄存器指针都将归零，RDFULL 位为低。
6	RDEMPT	接收缓冲器空标志位。 当未接收到数据时，RDEMPT 位为高，表明接收缓冲器为空。 当有接收到数据时，会存入接收缓冲器，RDEMPT 位为低，表明接收缓冲器为非空，此时 MCU 可通过访问 SPDR 寄存器来读取接收缓冲器中的数

		据。为确保所接收到的数据不会丢失，软件可在接收缓冲器为非空状态即 RDEMPT 位为低时读走接收缓冲器中的数据。 当对 RDEMPT 位进行置位操作（写 1）时，接收缓冲器地址将归零。 当同时对 RDEMPT 位和 WREMPT 位进行置位操作时，接收和发送缓冲器地址以及 SPI 移位寄存器指针都将归零，RDEMPT 位为高。
5	RDPTR1	接收缓冲器地址高位。
4	RDPTR0	接收缓冲器地址低位。 当对 SPDR 寄存器进行读操作时，MCU 将会从接收缓冲器中读到所接收的数据，同时接收缓冲器地址会进行累加。 当对 RDEMPT 位进行置位操作（写 1）时，接收缓冲器地址将归零。
3	WRFULL	发送缓冲器满标志位。 当发送缓冲器中的数据达到四个字节时，WRFULL 位为高，表明发送缓冲器为满。 当发送缓冲器中的数据少于四个字节时，WRFULL 位为低，表明发送缓冲器为非满。若想提高传输速度，软件可在发送缓冲器为非满状态即 WRFULL 位为低时写入数据，SPI 控制器会依次把数据发送出去。
2	WREMPT	发送缓冲器空标志位。 当写入发送缓冲器的数据均已发送完毕时，WREMPT 位为高，表明发送缓冲器为空，同时会置位中断标志位 SPIF。 当对 SPDR 寄存器进行写操作后，发送缓冲器地址会累加，写入发送缓冲器的数据未被全部发送时，接收缓冲器中至少有一个字节的数据，WREMPT 位为低，表明发送缓冲器非空。 当对 WREMPT 位进行置位操作（写 1）时，发送缓冲器地址将归零。 当同时对 RDEMPT 位和 WREMPT 位进行置位操作时，接收和发送缓冲器地址以及 SPI 移位寄存器指针都将归零，WREMPT 位为高。
1	WRPTR1	发送缓冲器地址高位。
0	WRPTR0	发送缓冲器地址低位。 当对 SPDR 寄存器进行写操作时，SPDR 中的数据将会被写入发送缓冲器，同时发送缓冲器地址会进行累加。 当对 WREMPT 位进行置位操作（写 1）时，发送缓冲器地址将归零。

USART0 - 通用同步/异步串行收发器

- 全双工操作（独立的串行接收和发送寄存器）
- 异步或同步操作
- 主机或从机操作
- 高精度的波特率发生器
- 支持 5，6，7，8，或 9 个数据位和 1，或 2 个停止位
- 硬件支持的奇偶产生和校验机制
- 数据过速检测
- 帧错误检测
- 噪声滤波，包括错误的起始位检测以及数字低通滤波器
- 三个独立的中断：发送结束中断，发送数据寄存器空中断以及接收结束中断
- 多处理器通信模式
- 倍速异步通信模式

综述



USART 结构图

USART 主要包括三个部分：时钟发生器，发送器和接收器。控制和状态寄存器由这三个部分共享。时钟发生器由波特率发生器和同步从机操作模式下外部输入时钟的同步逻辑组成。XCK 引脚只用于同步传输模式。发送器包括一个写数据缓冲器，串行移位寄存器，奇偶发生器以及处理不同帧格式所需的控制逻辑。写数据缓冲器允许连续发送数据而不会在数据帧之间引入延迟。接收器具有时钟和数据恢复单元，用于异步数据的接收。除了恢复单元，接收器还包括奇偶校验，控制逻辑，串行移位寄存器和一个两级接收缓冲器 UDR。接收器支持与发送器相同的帧格式，而且可以检测帧错误，数据过速和奇偶校验错误。

时钟产生

时钟产生逻辑为发送器和接收器产生基础时钟。USART 支持 4 种模式的时钟：正常的异步模式，倍速的异步模式，主机同步模式，以及从机同步模式。USCRC 的 UMSEL 位用于选择同步或异步模式。USCRA 的 U2X 位控制异步模式下的倍速使能。仅在同步模式下有效的 XCK 引脚的数据方向寄存器(与 IO 复用)决定了时钟源是由内部产生(主机模式)还是外部产生(从机模式)。

波特率发生器

波特率寄存器 UBRR 和降序计数器连接在一起作为 USART 的可编程的预分频器或波特率发生器。降序计数器工作在系统时钟(f_{sys})下，当其计数到零或 UBRRL 寄存器被写时，会自动加载 UBRR 寄存器的值。当计数到零时产生一个时钟，该时钟作为波特率发生器的输出时钟，频率为 f_{sys}/(UBRR+1)。

下表给出了各种工作模式下计算波特率（位/秒）以及 UBRR 值的公式。

工作模式	波特率计算公式(1)	UBRR 值计算公式
异步正常模式	BAUD = f_{sys}/(16*(UBRR+1))	UBRR = f_{sys}/(16*BAUD) - 1
异步倍速模式	BAUD = f_{sys}/(8*(UBRR+1))	UBRR = f_{sys}/(8*BAUD) - 1
同步主机模式	BAUD = f_{sys}/(2*(UBRR+1))	UBRR = f_{sys}/(2*BAUD) - 1

说明：

1. 波特率定义为每秒的位传输速度（bps）；
2. BUAD 为波特率，f_{sys} 为系统时钟，UBRR 为波特率寄存器 UBRRH 和 UBRRL 的组合值。

倍速工作模式

通过设定 UCSRA 寄存器的 U2X 位可以是传输速率加倍，该位只在异步工作模式下有效，同步工作模式下置该位为“0”。

设置该位将会把波特率分频器的分频值减半，有效地加倍异步通信的传输速率。在这种情况下，接收器只使用一半的采样数来对数据进行采样及时钟恢复，因此需要更精准的波特率设置和系统时钟。发送器则没有变化。

外部时钟

同步从机操作模式由外部时钟驱动。外部时钟经过同步寄存器和边沿检测器之后才被发送器

和接收器使用，这一过程会引入两个系统时钟的延时，因此外部 XCK 的最大时钟频率由以下公式限制：

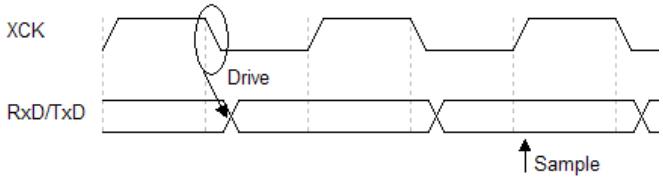
$$f_{XCK} < f_{sys}/4$$

要注意 fsys 有系统时钟的稳定性决定，为了防止因频率漂移而丢失数据，建议保留足够的裕量。

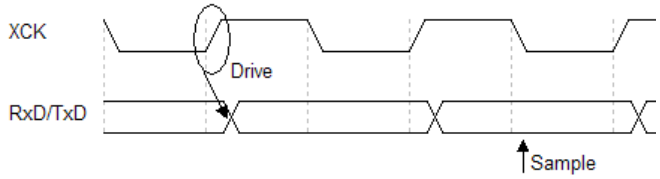
同步时钟操作

同步模式下，XCK 引脚被用于时钟输入（从机模式）或时钟输出（主机模式）。时钟的边沿与数据采样和数据变化关系的基本规律是：对数据输入端（RxD）采样所使用的时钟沿与数据输出端变化所使用的时钟沿是相反的。

UCPOL = 1



UCPOL = 0



同步模式下的 XCK 时序

如上图所示，当 UCPOL 值为“1”时，在 XCK 的下降沿改变数据输出，在 XCK 的上升沿进行数据采样；当 UCPOL 值为“0”时，在 XCK 的上升沿改变数据输出，在 XCK 的下降沿进行数据采样。

帧格式

一个串行数据帧由数据字加上同步位（起始位和停止位）以及用于纠错的奇偶校验位构成。USART 接受以下 30 种组合的数据帧格式：

- 1 个起始位
- 5、6、7、8 或 9 个数据位
- 无校验位、奇校验位或偶校验位
- 1 或 2 个停止位

数据帧以起始位开始，紧接着是数据字的最低位，接着是其它数据位，以数据字的最高位结束，最多成功传输 9 位数据。如果使能了校验，校验位将紧接着数据字，最后是停止位。当一个完整的数据帧传输后，可以立即传输下一个新的数据帧，或者使传输线处于空闲（高电平）状态。下图为可能的数据帧结构，方括号中的位是可选的。



USART 帧结构图

说明：

1）IDLE 通信线（RxD 或 TxD）上没有数据传输，线路空闲时必须为高电平

2）St 起始位，总是为低电平

3）0-8 数据位

4）P 校验位，奇校验或偶校验

5）Sp 停止位，总是为高电平

数据帧的结构由 UCSRB 和 UCSRC 寄存器中的 UCSZ[2:0]、UPM[1:0]和 USBS 设定。接收与发送使用相同的设置。设置的任何改变都可能破坏正在进行的数据传输。其中，UCSZ[2:0]确定了数据帧的数据位数，UPM[1:0]用于使能和确定校验的类型，USBS 设置帧有一位或两位结束位。接收器会忽略第二个停止位，因此帧错误只在第一个结束位为“0”时被检测到。

校验位计算

校验位的计算是对数据的各个位进行异或运算。如果选择了奇校验，则异或结果还需要取反。校验位与数据位的关系如下：

$$P_{even} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 0$$

$$P_{odd} = d_{n-1} \oplus \ldots \oplus d_3 \oplus d_2 \oplus d_1 \oplus d_0 \oplus 1$$

说明：

1）P_{even} 偶校验结果

2）P_{odd} 奇校验结果

3）d_n 第 n 个数据位

USART 初始化

进行通信之前首先要对 USART 进行初始化。初始化过程通常包括波特率的设定，帧结构的设定，以及根据需要使能接收器或发送器。对于中断驱动的 USART 操作，在初始化时要清零全局中断标志并禁止 USART 的所有中断。

在进行重新初始化比如改变波特率或帧结构时，必须确保没有数据传输。TXC 标志位可以用来检测发送器是否完成了所有传输，RXC 标志位可以用来检测接收缓冲器中是否还有数据未被读出。如果 TXC 标志位用作此用途，在每次发送数据之前（写 UDR 寄存器之前）必须清零 TXC 标志位。

发送器

置位 UCSRB 寄存器的 TXEN 位将使能 USART 的数据发送。使能后 TxD 引脚的通用 IO 功能即被 USART 功能所取代，成为发送器的串行输出。发送数据之前要设置好波特率、工作模式与帧格式。如果使用同步发送模式，施加于 XCK 引脚上的时钟信号即为数据发送的时钟。

发送 5 到 8 为数据的帧

将需要发送的数据加载到发送缓冲器中来启动数据发送。CPU 通过写 UDR 寄存器来加载数据。当发送移位寄存器可以发送新一帧数据的时候，缓冲器中的数据将转移到移位寄存器中。当移位寄存器处于空闲状态（没有正在进行的数据传输），或者前一帧数据的最后一个停止位发送完毕，它将加载新的数据。一旦移位寄存器加载了新的数据，它将按照既定的设置传

输一个完整的帧。

发送 9 位数据的帧

如果发送 9 位数据的帧，应先将数据的第 9 位写入寄存器 UCSRB 的 TXB8 位，然后再将低 8 位数据写入发送数据寄存器 UDR。第 9 位数据在多机通信中用于表示地址帧，在同步通信中可以用于协议处理。

发送奇偶校验位

奇偶校验产生电路为串行数据帧生成相应的校验位。当校验位使能时（UPM1 = 1），发送控制逻辑电路会在数据字的最后一位与第一个停止位之间插入奇偶校验位。

发送标志位与中断处理

USART 发送器有两个标志位：USART 数据寄存器空标志 UDRE 和传输结束标志 TXC，两个标志位都可以产生中断。

数据寄存器空标志 UDRE 用来表示发送缓冲器是否可以写入一个新的数据。该位在发送缓冲器空时被置“1”，满时被置“0”。当 UDRE 位为“1”时，CPU 可以往数据寄存器 UDR 写入新的数据，反之则不能。

当 UCSRB 寄存器中的数据寄存器空中断使能位 UDRIE 为“1”时，只要 UDRE 被置位（且全局中断使能），就将产生 USART 数据寄存器空中断请求。对寄存器 UDR 执行写操作将清零 UDRE。当采用中断方式传输数据时，在数据寄存器空中断服务程序中必须写入一个新的数据到 UDR 以清零 UDRE，或者是禁止数据寄存器空中断。否则一旦该中断服务程序结束，一个新的中断将再次产生。

当整个数据帧被移出发送移位寄存器，同时发送寄存器中又没有新的数据时，发送结束标志 TXC 将被置位。当 UCSRB 上的发送结束中断使能位 TXCIE（且全局中断使能）置“1”时，随着 TXC 标志位被置位，USART 发送结束中断将被执行。一旦进入中断服务程序，TXC 标志位即被自动清零，CPU 也可以对该位写“1”来清零。

禁止发送器

当 TXEN 清零后，只有等所有的数据都发送完成以后发送器才能够真正禁止，即发送移位寄存器与发送缓冲寄存器中都没有要传送的数据。发送器禁止以后，TxD 引脚恢复其通用 IO 功能。

接收器

置位 UCSRB 寄存器的接收允许位（RXEN）即可启动 USART 接收器。使能后 RxD 引脚的通用 IO 功能被 USART 功能所取代，成为接收器的串行输入口。进行数据接收之前首先要设置好波特率、操作模式及帧格式。如果使用同步接收模式，XCK 引脚上的时钟被用为传输时钟。

接收 5 到 8 位数据的帧

一旦接收器检测到一个有效的起始位，便开始接受数据。起始位后的每一位数据都将以所设定的波特率或 XCK 时钟来进行接收，直到收到一帧数据的第一个停止位，第二个停止位会被

接收器忽略。接收到的每一位数据被送入接收移位寄存器，收到第一个停止位以后，接收器置位位于 UCSRA 寄存器的接收数据完成标志 RXC 位，并把移位寄存器中完整的数据帧转移到接收缓冲器中，CPU 通过读取 UDR 寄存器就可以获得接收到的数据。

接收 9 位数据的帧

如果设定了 9 位数据的数据帧，在从 UDR 读取低 8 位数据之前必须首先读取寄存器 UCSRB 的 RXB8 位来获得第 9 位数据。这个规则同样适用于状态标志位 FE、DOR 以及 PE。读取 UDR 存储单元会改变接收缓冲器的状态，进而改变同样存储于缓冲器中的 TXB8、FE、DOR 及 PE 位。

接收结束标志及中断处理

USART 接收器有一个标志位：接收结束标志 RXC，用来表明接收缓冲器中是否有未被读出的数据。当接收缓冲器中有未被读出的数据时，此位为“1”，反之为“0”。如果接收器被禁止，接收缓冲器会被刷新，RXC 也会被清零。

置位 UCSRB 的接收结束中断使能位 RXCIE 后，只要 RXC 标志被置位（且全局中断被使能），就会产生 USART 接收结束中断。使用中断方式进行数据接收时，数据接收结束中断服务程序必须从 UDR 读取数据来清零 RXC 标志，否则只要中断处理程序一结束，一个新的中断就会产生。

接收错误标志

USART 接收器有三个错误标志：帧错误 FE、数据溢出 DOR 及奇偶校验错误 PE。它们都位于 UCSRA 寄存器。错误标志与数据帧一起保存在接收缓冲器当中。所有的错误标志都不能产生中断。

帧错误标志 FE 表明存储在接收缓冲器中的下一个可读帧的第一个停止位的状态。停止位正确（值为“1”）则 FE 标志为“0”，否则 FE 标志为“1”。这个标志可用来检测同步丢失、传输中断，也可用于协议处理。

数据溢出标志 DOR 表明由于接收缓冲器满造成了数据丢失。当接收缓冲器为满，接收移位寄存器中已有数据，若此时检测到一个新的起始位，数据溢出就产生了。DOR 标志被置位即表明在最近一次读取 UDR 和下一次读取 UDR 之间丢失了一个或多个数据帧。当数据帧成功地从移位寄存器转入接收缓冲器后，DOR 标志被清零。

奇偶校验错标志 PE 表明接收缓冲器中的下一帧数据在接收时有奇偶错误。如果不使能奇偶校验，PE 被清零。

奇偶校验器

置位奇偶校验模式位 UPM1 将启动奇偶校验器。校验的模式（偶校验或奇校验）由 UPM0 决定。奇偶校验使能后，校验器将计算输入数据的奇偶并把结果与数据帧的奇偶位进行比较。校验结果将与数据和停止位一起存储在接收缓冲器中。CPU 通过读取 PE 位来检查接收的帧当中是否有奇偶错误。如果下一个从接收缓冲器中读出的数据有奇偶错误，并且奇偶校验使能，则 UPE 被置位，一直有效到接收缓冲器 UDR 被读取。

禁止接收器

与发送器相比，禁止接收器即刻起作用。正在接收的数据将丢失。禁止接收器（RXEN 清零）后，接收器将不再占用 RxD 引脚，接收缓冲器也会被刷新。

异步数据接收

USART 有一个时钟恢复单元和数据恢复单元来处理异步数据接收。时钟恢复逻辑用于同步从 RxD 引脚输入的异步串行数据和内部的波特率时钟。数据恢复逻辑用于采集数据，并通过低通滤波器过滤所输入的每一位数据，从而提高接收器的抗干扰性能。异步接收的工作范围依赖于内部波特率时钟的精度、帧输入的速率及一帧所包含的数据位数。

异步工作范围

接收器的工作范围依赖于接收到的数据速率与内部波特率之间的不匹配程度。如果发送器以过快或过慢的比特率传输数据，或者接收器内部产生的波特率没有相同的频率，那么接收器就无法与起始位同步。为了确保接收器不会错过下一帧起始位的采样，数据输入速率和内部接收器波特率不能相差太大，用它们之间的比值来描述波特率的误差范围。下面两个表格分别给出了普通模式下和倍速模式下容许的最大波特率误差范围。

普通模式下最大接收器波特率误差范围

数据位+奇偶位长度和	最大误差范围（%）	推荐误差范围（%）
5	+6.7/-6.8	±3.0
6	+5.8/-5.9	±2.5
7	+5.1/-5.2	±2.0
8	+4.6/-4.5	±3.0
9	+4.1/-4.2	±1.5
10	+3.8/-3.8	±1.5

倍速模式下最大接收器波特率误差范围

数据位+奇偶位长度和	最大误差范围（%）	推荐误差范围（%）
5	+5.7/-5.9	±2.5
6	+4.9/-5.1	±2.0
7	+4.4/-4.5	±1.5
8	+3.9/-4.0	±1.5
9	+3.5/-3.6	±1.0
10	+3.2/-3.3	±1.0

从表中可以看出，普通模式下波特率允许有更大的变化范围。上述推荐的波特率误差范围是假定接收器和发送器对最大总误差具有同等贡献的前提下得出的。产生接收器波特率误差的可能原因有两个。首先，接收器系统时钟的稳定性与工作电压和温度有关。使用晶振来产生系统时钟时一般不会有此问题，但使用内部振荡器时，系统时钟可能会有偏差。第二个原因是波特率发生器不一定能通过对系统时钟的分频来得到恰好想要的波特率。此时可以调整 UBRR 的值，使得误差低至可以接受。

波特率设置及引入误差

对于标准晶振及谐振器频率来说，异步模式下的实际通信的波特率可通过波特率计算公式来获得，它与常用通信波特率之间的误差可用如下公式来计算：

$$Error[\%] = (Baud_{real}/Baud - 1)*100\%$$

其中，Baud 为常用的通信波特率，$Baud_{real}$ 为通过计算公式算出来的波特率，带入波特率计算公式即可得到波特率误差与系统时钟 f_{sys} 和波特率寄存器 UBRR 值之间的关系如下：

普通模式：

$$Error[\%] = (f_{sys}/(16*(UBRR+1))/Baud - 1)*100\%$$

倍速模式：

$$Error[\%] = (f_{sys}/(8*(UBRR+1))/Baud - 1)*100\%$$

当不考虑通信两边的时钟误差，即系统时钟 f_{sys} 为标准时钟时，即可得到波特率误差 UBRR 值之间的关系。下表即为 16MHz 系统时钟下不同 UBRR 值设置下的波特率误差。

16MHz 系统时钟下设置 UBRR 值所产生的误差

波特率 (bps)	f_{sys} = 16.000MHz			
	普通模式(U2X = 0)		倍速模式(U2X = 1)	
	UBRR	误差	UBRR	误差
2400	416	-0.1%	832	0.0%
4800	207	0.2%	416	-0.1%
9600	103	0.2%	207	0.2%
14.4K	68	0.6%	138	-0.1%
19.2K	51	0.2%	103	0.2%
28.8K	34	-0.8%	68	0.6%
38.4K	25	2.1%	34	-0.8%
57.6K	16	0.2%	51	0.2%
76.8K	12	0.2%	25	0.2%
115.2K	8	-3.5%	16	2.1%
230.4K	3	8.5%	8	-3.5%
250K	3	0%	7	0%
0.5M	1	0%	3	0%
1M	0	0%	1	0%

多处理器通信模式

置位 UCSRA 的多处理器通信模式(MPCM)位可以对 USART 接收器接收到的数据帧进行过滤。那些没有地址信息的帧将被忽略，也不会存入接收缓冲器。在一个多处理器系统中，各处理器通过相同的串行总线进行通信，这种过滤有效的减少了需要 CPU 处理的数据帧的数量。MPCM 位的设置不影响发送器的工作，但在多处理器通信的系统中，它的使用方法会有所不同。

如果接收器所接收的数据帧长度为 5 到 8 位，那么第一个停止位会用来表示当前帧包含的是数据还是地址信息。如果接收器所接收的数据帧长度是 9 位，那么由第 9 位来确定是数据还是地址信息。如果帧类型标志位为“1”，那么这是地址帧，否则为数据帧。

在多处理器通信模式下，允许多个从处理器从一个主处理器接收数据。首先要通过解码地址帧来确定所寻址的是哪一个从处理器。被寻址的从处理器将正常接收后续的数据，而其他的从处理器则会忽略这些数据帧直到接收到下一个地址帧。

对于一个作为主机的处理器来说，它可以使用 9 位数据帧格式，并用第 9 位数据来标识帧格式。在这种通信模式下，从处理器也必须工作于 9 位数据帧格式。
下面即为多处理器通信模式下进行数据交换的步骤：

1. 所有从处理器都工作在多处理器通信模式（置位 MPCM）；
2. 主处理器发送地址帧，所有从处理器都接收此帧。从处理器 UCSRA 寄存器的 RXC 位正常置位；
3. 每个从处理器都读取 UDR 寄存器的内容，解码地址帧来确定是否被选中。如果选中，就清零 UCSRA 寄存器的 MPCM 位，未被选中，则保持 MPCM 为“1”并等待下一个地址帧的到来；
4. 被寻址的从处理器接收所有的数据帧，直到收到一个新的地址帧。未被寻址的从处理器忽略这些数据帧；
5. 被寻址的从处理器收到最后一个数据帧后，置位 MPCM 位，并等待下一个地址帧的到来。然后从第二步骤重复进行。

使用 5 到 8 位数据的帧格式是可以的，但是不切实际，因为接收器必须在使用 n 和 n+1 帧格式之间进行切换。由于接收器和发送器使用相同的字符长度设置，这种设置使得全双工操作变得很困难。如果使用 5 到 8 位数据的帧格式，发送器应该设置两个停止位，其中第一个停止位被用于判断帧类型。

寄存器定义

UCSRA – USART 控制和状态寄存器 A

UCSRA – USART 控制和状态寄存器 A								
地址: 0xC0					默认值: 0x20			
Bit	7	6	5	4	3	2	1	0
Name	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPME
R/W	R	R/W	R	R	R	R	R/W	R/W

Bit	Name	描述
7	RXC	接收结束标志位。 当 RXC 的值为“1”时，表明接收缓冲器中有未读出的数据。当 RXC 的值为“0”时，表明接收缓冲器中没有未读出的数据。接收器禁止时，接收缓冲器被刷新，导致 RXC 被清零。当接收结束中断使能位 RXCIE 为“1”时，RXC 可用来产生接收结束中断。
6	TXC	发送结束标志位。 发送移位寄存器中的数据被送出，且发送缓冲器为空时 TXC 置位。执行发送结束中断时 TXC 自动清零，也可以通过对 TXC 写“1”来进行清零。当发送结束中断使能位 TXCIE 为“1”时，TXC 可用来产生发送结束中断。

5	UDRE	数据寄存器空标志位。 当 UDRE 为“1”时，表明 USART 发送数据缓冲器为空，可以写入数据。 当 UDRE 为“0”时，表明 USART 发送数据缓冲器为满，不能写入数据。 当数据寄存器空中断使能位 UDRIE 为“1”时，UDRE 可用来产生数据寄存器空中断。
4	FE	帧错误标志位。 当 FE 为“1”时，表明接收数据缓冲器接收到的数据有帧错误，即第一个停止位为“0”。当 FE 为“0”时，表明接收数据缓冲器接收到的数据没有帧错误，即第一个停止位为“1”。FE 被置位后会一直有效到 UDR 被读取。对 UCSRA 进行写入时，FE 这一位要写“0”。
3	DOR	数据溢出标志位。 当接收缓冲器为满（包含了两个数据），接收移位寄存器中有数据，若此时检测到一个新的起始位，数据溢出产生，DOR 被置位，一直有效到 UDR 被读取。对 UCSRA 进行写入时，DOR 这一位要写“0”。
2	PE	奇偶校验错误标志位。 当奇偶校验使能（UPM1 为“1”）时，且接收缓冲器中所接收到的数据帧有奇偶校验错误，PE 被置位，一直有效到 UDR 被读取。对 UCSRA 进行写入时，PE 这一位要写“0”。
1	U2X	倍速发送使能位。 当 U2X 为“1”时，异步通信模式的传输速率加倍。当 U2X 为“0”时，异步通信模式的传输速率为普通速率。 这一位仅在异步操作模式下有效，使用同步操作模式时将此位清零。
0	MPCM	多处理器通信模式使能位。 设置 MPCM 位将启动多处理器通信模式。MPCM 置位后，USART 接收器接收到的那些不包含地址信息的输入帧都将被忽略。发送器不受 MPCM 设置的影响。

UCSRB – USART 控制和状态寄存器 B

UCSRB – USART 控制和状态寄存器 B								
地址: 0xC1					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W

Bit	Name	描述
7	RXCIE	接收结束中断使能位。 置位后使能 RXC 中断，清零后禁止 RXC 中断。当 RXCIE 为“1”，全局中断使能，UCSRA 寄存器的 RXC 为“1”时可以产生 USART 接收结束中断。
6	TXCIE	发送结束中断使能位。 置位后使能 TXC 中断，清零后禁止 TXC 中断。当 TXCIE 为“1”，全局中断使能，UCSRA 寄存器的 TXC 为“1”时可以产生 USART 发送结束中断。
5	UDRIE	数据寄存器空中断使能位。 置位后使能 UDRE 中断，清零后禁止 UDRE 中断。当 UDRIE 为“1”，全局中断使能，UCSRA 寄存器的 UDRE 为“1”时可以产生 USART 数据寄存器空

		中断。
4	RXEN	接收使能位。 置位后启动 USART 接收器。RxD 引脚的通用 IO 功能被 USART 接收所取代。禁止接收器将刷新接收缓冲器，并使 FE、DOR 及 PE 标志无效。
3	TXEN	发送使能位。 置位后启动 USART 发送器。TxD 引脚的通用 IO 功能被 USART 发送所取代。TXEN 清零后，只有等到所有的数据发送完成后才能够真正禁止 USART 发送。
2	UCSZ2	字符长度控制第 2 位。 UCSZ2 与 UCSRC 寄存器的 UCSZ1:0 结合在一起设置数据帧所包含的数据位数。
1	RXB8	接收数据第 8 位。 当数据帧长度为 9 位时，RXB8 是接收数据的最高位。读取 UDR 所包含的低 8 位数据之前要先读取 RXB8。
0	TXB8	发送数据第 8 位。 当数据帧长度为 9 位时，TXB8 是发送数据的最高位。写入 UDR 所包含的低 8 位数据之前要先写入 TXB8。

UCSRC– USART 控制和状态寄存器 C

UCSRC– USART 控制和状态寄存器 C								
地址: 0xC2					默认值: 0x06			
Bit	7	6	5	4	3	2	1	0
Name	UMSEL1	UMSEL0	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述	
7:6	UMSEL1:0	USART 模式选择位。 UMSEL 选择同步或异步操作模式。	
		UMSEL	模式
		0	USART 异步操作模式
		1	USART 同步操作模式
		2	SPI 从机操作模式
		3	SPI 主机操作模式
5:4	UPM1:0	奇偶校验模式选择位。 高位 UPM1 选择使能或禁止奇偶校验，低位 UPM0 选择奇校验或偶校验。	
		UPM1:0	模式
		0	禁止奇偶校验
		1	保留
		2	使能偶校验
		3	使能奇校验
3	USBS	停止位选择位。选择停止位的位数。	
		USBS	停止位位数
		0	1

		1	2
2:1	UCSZ1:0	数据帧字符长度选择位。 UCSZ1:0 与 UCSRB 寄存器的 UCSZ2 结合起来设置数据帧包含的数据位数。	
		UCSZ2:0	数据帧长度
		0	5 位
		1	6 位
		2	7 位
		3	8 位
		4	保留
		5	保留
		6	保留
		7	9 位
0	UCPOL	时钟极性选择位。 在 USART 同步工作模式下，UCPOL 设置了输出数据的改变和输入数据的采样与同步时钟 XCK 之间的关系。使用异步工作模式下与 UCPOL 无关，将这一位清零	
		UCPOL	发送数据改变	接收数据采样
		0	XCK 的上升沿	XCK 的下降沿
		1	XCK 的下降沿	XCK 的上升沿

UBRRL – USART 波特率寄存器低字节

UBRRL – USART 波特率寄存器低字节								
地址: 0xC4					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UBRR7	UBRR6	UBRR5	UBRR4	UBRR3	UBRR2	UBRR1	UBRR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	UBRR[7:0]	USART 波特率寄存器的低字节部分。 USART 波特率寄存器包含 UBRRL 和 UBRRH 两部分，结合在一起用来设置通信的波特率。

UBRRH – USART 波特率寄存器高字节

UBRRH – USART 波特率寄存器高字节								
地址: 0xC5					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	UBRR11	UBRR10	UBRR9	UBRR8
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	描述
7:4	-	保留。

<table>
<tr><td rowspan="4">3:0</td><td rowspan="4">UBRR[11:8]</td><td colspan="2">USART 波特率寄存器的高字节部分。
USART 波特率寄存器包含 UBRRL 和 UBRRH 两部分，结合在一起用来设置通信的波特率。
UBRR = {UBRR[11:8], UBRRL}</td></tr>
<tr><td>工作模式</td><td>波特率计算公式</td></tr>
<tr><td>异步正常模式</td><td>BAUD = f_{sys}/(16*(UBRR+1))</td></tr>
<tr><td>异步倍速模式</td><td>BAUD = f_{sys}/(8*(UBRR+1))</td></tr>
<tr><td></td><td></td><td>同步主机模式</td><td>BAUD = f_{sys}/(2*(UBRR+1))</td></tr>
</table>

UDR – USART 数据寄存器

UDR – USART 数据寄存器								
地址: 0xC6					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UDR7	UDR6	UDR5	UDR4	UDR3	UDR2	UDR1	UDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

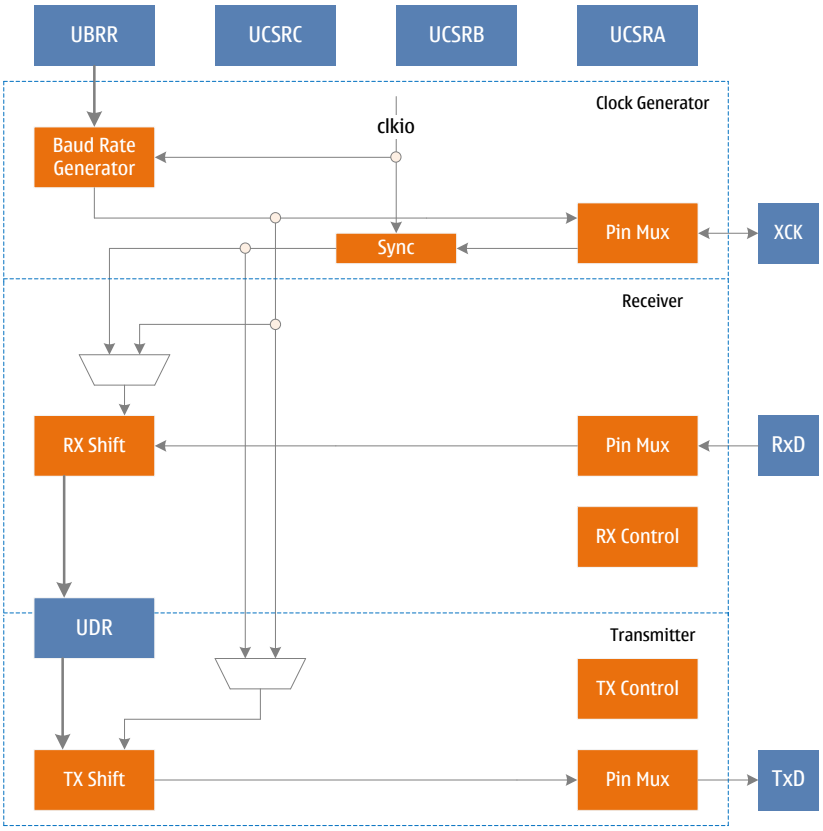
Bit	Name	描述
7:0	UDR	USART 发送和接收的数据。 USART 发送数据缓冲器和接收数据缓冲器共享 USART 数据寄存器 UDR。将数据写入 UDR 即写入发送数据缓冲器，从 UDR 读取数据即读取接收数据缓冲器。 在 5 到 8 位数据帧模式下，未使用的第 9 位被发送器忽略，而接收器则将它们设置为 0。 只有当 UCSRA 寄存器的 UDRE 标志为“1”时才能对发送缓冲器进行写操作，否则发送器的操作会出错。当发送移位寄存器为空时，发送器会把发送缓冲器中的数据加载到发送移位寄存器中，然后数据串行地从 TxD 引脚输出。 接收缓冲器包含一个两级 FIFO，一旦接收缓冲器被读取，FIFO 就会改变它的状态。

USART0 - SPI 工作模式

- 全双工操作，三线同步数据传输
- 主机或从机操作
- 支持全部四种工作模式（模式 0、1、2 和 3）
- 低位或高位首先传输（可配置的数据传输顺序）
- 队列操作（双缓冲器）
- 高分辨率波特率产生器

综述

当设置 USCRC 的 UMSEL1 位为“1”时，使能 SPI 工作模式，用 USPI 来表示。此 SPI 模块为三线 SPI 工作模式，与四线 SPI 模式相比，缺少从机选择线，其它三根线均一致。USPI 占用 USART 的资源，包括发送和接收移位寄存器和缓冲器，以及波特率发生器。奇偶校验产生和检查逻辑，数据和时钟恢复逻辑均无效。控制和状态寄存器的地址是一样的，不过寄存器位的功能会随着 SPI 工作模式的需要而发生改变。



USART in SPI 结构图

时钟产生

当 SPI 工作在主机模式时，需要提供通信用的时钟，复用 USART 的波特率发生器来产生这个时钟。该时钟从 XCK 引脚输出，因此 XCK 引脚的数据方向寄存器（DDR_XCK）必须设置为

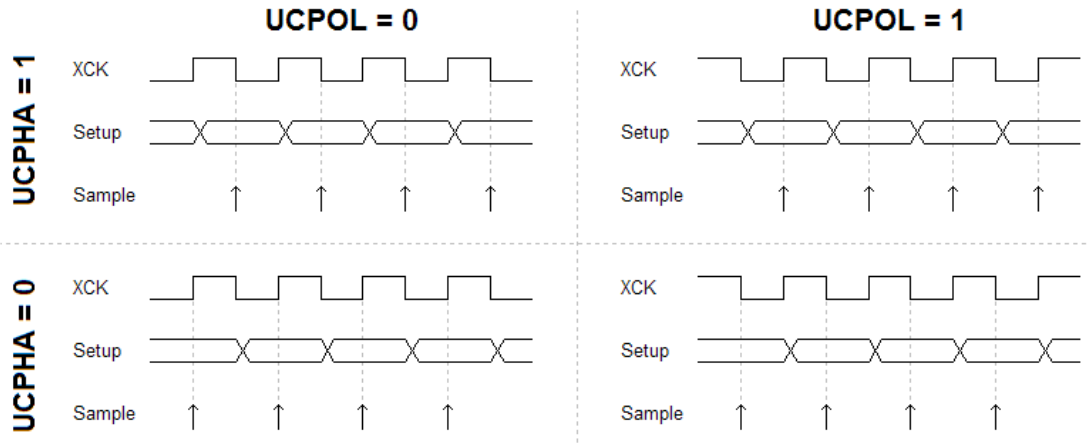
“1”。
时钟频率有以下计算公式决定：
BAUD = f_{sys}/(2*(UBRR+1))
当 SPI 工作在从机模式时，通信时钟由外部主机提供，从 XCK 引脚输入，因此 XCK 引脚的数据方向寄存器（DDR_XCK）必须设置为“0”。

SPI 数据模式和时序

SPI 有四种时钟相位和极性的组合方式，有控制位 UCPHA 和 UCPOL 来决定，具体的控制如下表和下图所示：

SPI 工作模式

SPI 模式	UCPOL	UCPHA	起始沿	结束沿
0	0	0	上升沿采样	下降沿设置
1	0	1	上升沿设置	下降沿采样
2	1	0	下降沿采样	上升沿设置
3	1	1	下降沿设置	上升沿采样



SPI 工作模式图示

帧格式

SPI 的一个串行帧可以由最低位或最高位开始，到最高位或最低位结束，总共 8 位数据。一帧结束以后，可以紧接着传输新的一帧，传输结束即可拉高数据线为空闲状态。

数据传输

SPI 置 UCSRB 寄存器的 TXEN 位为“1”来使能发送器，TxD 引脚被发送器占用来发送串行输出数据。此时接收器可以不使能。
SPI 置 UCSRB 寄存器的 RXEN 位为“1”来使能接收器，RxD 引脚被接收器占用来接收串行输入数据。此时发送器须使能。
SPI 发送和接收都使用 XCK 来当作传输时钟。

进行通信之前首先要对 SPI 进行初始化。初始化过程通常包括波特率的设定，帧数据位传输顺序的设定，以及根据需要使能接收器或发送器。对于中断驱动的 SPI 操作，在初始化

时要清零全局中断标志并禁止 SPI 的所有中断。
在进行重新初始化比如改变波特率或帧结构时，必须确保没有数据传输。TXC 标志位可以用来检测发送器是否完成了所有传输，RXC 标志位可以用来检测接收缓冲器中是否还有数据未被读出。如果 TXC 标志位用作此用途，在每次发送数据之前（写 UDR 寄存器之前）必须清零 TXC 标志位。

初始化 SPI 以后，往 UDR 寄存器写入数据即可开始数据传输。由于发送器控制着传输时钟，发送和接收数据均是如此操作。当发送移位寄存器准备好发送新一帧数据的时候，发送器就会把写入到 UDR 寄存器的数据从发送缓冲器移到发送移位寄存器里并发送出去。为了保证输入缓冲器和发送数据同步，每发送一个字节的数据后都必须读取一次 UDR 寄存器。当发生数据溢出时，最近收到的数据将会丢失，而不是最早收到的数据。

发送标志位与中断

SPI 发送器有两个标志位：SPI 数据寄存器空标志 UDRE 和传输结束标志 TXC，两个标志位都可以产生中断。

数据寄存器空标志 UDRE 用来表示发送缓冲器是否可以写入一个新的数据。该位在发送缓冲器空时被置“1”，满时被置“0”。当 UDRE 位为“1”时，CPU 可以往数据寄存器 UDR 写入新的数据，反之则不能。

当 UCSRB 寄存器中的数据寄存器空中断使能位 UDRIE 为“1”时，只要 UDRE 被置位（且全局中断使能），就将产生 SPI 数据寄存器空中断请求。对寄存器 UDR 执行写操作将清零 UDRE。当采用中断方式传输数据时，在数据寄存器空中断服务程序中必须写入一个新的数据到 UDR 以清零 UDRE，或者是禁止数据寄存器空中断。否则一旦该中断服务程序结束，一个新的中断将再次产生。

当整个数据帧被移出发送移位寄存器，同时发送寄存器中又没有新的数据时，发送结束标志 TXC 将被置位。当 UCSRB 上的发送结束中断使能位 TXCIE（且全局中断使能）置“1”时，随着 TXC 标志位被置位，SPI 发送结束中断将被执行。一旦进入中断服务程序，TXC 标志位即被自动清零，CPU 也可以对该位写“1”来清零。

禁止发送器

当 TXEN 清零后，只有等所有的数据都发送完成以后发送器才能够真正禁止，即发送移位寄存器与发送缓冲寄存器中都没有要传送的数据。发送器禁止以后，TxD 引脚恢复其通用 IO 功能。

接收结束标志及中断

SPI 接收器有一个标志位：接收结束标志 RXC，用来表明接收缓冲器中是否有未被读出的数据。当接收缓冲器中有未被读出的数据时，此位为“1”，反之为“0”。如果接收器被禁止，接收缓冲器会被刷新，RXC 也会被清零。置位 UCSRB 的接收结束中断使能位 RXCIE 后，只要 RXC 标志被置位（且全局中断被使能），就会产生 SPI 接收结束中断。使用中断方式进行数据接收时，数据接收结束中断服务程序必须从 UDR 读取数据来清零 RXC 标志，否则只要

中断处理程序一结束，一个新的中断就会产生。

禁止接收器

与发送器相比，禁止接收器即刻起作用。正在接收的数据将丢失。禁止接收器（RXEN 清零）后，接收器将不再占用 RxD 引脚，接收缓冲器也会被刷新。

寄存器定义

USART 寄存器列表

寄存器	地址	默认值	描述
UCSRA	0xC0	0x20	USPI 控制和状态寄存器 A
UCSRB	0xC1	0x00	USPI 控制和状态寄存器 B
UCSRC	0xC2	0x06	USPI 控制和状态寄存器 C
UBRRL	0xC4	0x0	USPI 波特率寄存器低字节
UBRRH	0xC5	0x0	USPI 波特率寄存器高字节
UDR	0xC6	0x0	USPI 数据寄存器

UCSRA – USPI 控制和状态寄存器 A

UCSRA – USPI 控制和状态寄存器 A								
地址: 0xC0					默认值: 0x20			
Bit	7	6	5	4	3	2	1	0
Name	RXC	TXC	UDRE	-	-	-	-	-
R/W	R	R/W	R	-	-	-	-	-

Bit	Name	描述
7	RXC	接收结束标志位。 当 RXC 的值为“1”时，表明接收缓冲器中有未读出的数据。当 RXC 的值为“0”时，表明接收缓冲器中没有未读出的数据。接收器禁止时，接收缓冲器被刷新，导致 RXC 被清零。当接收结束中断使能位 RXCIE 为“1”时，RXC 可用来产生接收结束中断。
6	TXC	发送结束标志位。 发送移位寄存器中的数据被送出，且发送缓冲器为空时 TXC 置位。执行发送结束中断时 TXC 自动清零，也可以通过对 TXC 写“1”来进行清零。当发送结束中断使能位 TXCIE 为“1”时，TXC 可用来产生发送结束中断。
5	UDRE	数据寄存器空标志位。 当 UDRE 为“1”时，表明 USPI 发送数据缓冲器为空，可以写入数据。当 UDRE 为“0”时，表明 USPI 发送数据缓冲器为满，不能写入数据。当数据寄存器空中断使能位 UDRIE 为“1”时，UDRE 可用来产生数据寄存器空中断。
4:0	-	USPI 下保留。

UCSRB – USPI 控制和状态寄存器 B

UCSRB – USPI 控制和状态寄存器 B								
地址: 0xC1					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	RXCIE	TXCIE	UDRIE	RXEN	TXEN	-	-	-
R/W	R/W	R/W	R/W	R/W	R/W	-	-	-

Bit	Name	描述
7	RXCIE	接收结束中断使能位。 置位后使能 RXC 中断，清零后禁止 RXC 中断。当 RXCIE 为“1”，全局中断使能，UCSRA 寄存器的 RXC 为“1”时可以产生 USPI 接收结束中断。
6	TXCIE	发送结束中断使能位。 置位后使能 TXC 中断，清零后禁止 TXC 中断。当 TXCIE 为“1”，全局中断使能，UCSRA 寄存器的 TXC 为“1”时可以产生 USPI 发送结束中断。
5	UDRIE	数据寄存器空中断使能位。 置位后使能 UDRE 中断，清零后禁止 UDRE 中断。当 UDRIE 为“1”，全局中断使能，UCSRA 寄存器的 UDRE 为“1”时可以产生 USPI 数据寄存器空中断。
4	RXEN	接收使能位。 置位后启动 USPI 接收器。RxD 引脚的通用 IO 功能被 USPI 接收所取代。禁止接收器将刷新接收缓冲器。
3	TXEN	发送使能位。 置位后启动 USPI 发送器。TxD 引脚的通用 IO 功能被 USPI 发送所取代。TXEN 清零后，只有等到所有的数据发送完成后才能够真正禁止 USART 发送。
2:0	-	USPI 下保留。

UCSRC– USART 控制和状态寄存器 C

UCSRC– USART 控制和状态寄存器 C								
地址: 0xC2					默认值: 0x86			
Bit	7	6	5	4	3	2	1	0
Name	UMSEL1	UMSEL0	-	-	-	DORD	UCPHA	UCPOL
R/W	R/W	R/W	-	-	-	R/W	R/W	R/W

Bit	Name	描述	
7:6	UMSEL1:0	USART 模式选择位。 UMSEL 选择同步或异步操作模式。	
		UMSEL	模式
		0	USART 异步操作模式
		1	USART 同步操作模式
		2	SPI 从机操作模式
		3	SPI 主机操作模式
5:3	-	USPI 下保留。	

<table>
<tr><td rowspan="4">2</td><td rowspan="4">DORD</td><td colspan="3">数据传输顺序选择位。</td></tr>
<tr><td>DORD</td><td colspan="2">数据顺序</td></tr>
<tr><td>0</td><td colspan="2">高位先传输</td></tr>
<tr><td>1</td><td colspan="2">低位先传输</td></tr>
<tr><td rowspan="4">1</td><td rowspan="4">UCPHA</td><td colspan="3">时钟相位选择。
UCPHA 选择数据采样发生在起始沿或结束沿。</td></tr>
<tr><td>UCPHA</td><td colspan="2">采样时刻</td></tr>
<tr><td>0</td><td colspan="2">起始沿</td></tr>
<tr><td>1</td><td colspan="2">结束沿</td></tr>
<tr><td rowspan="4">0</td><td rowspan="4">UCPOL</td><td colspan="3">时钟极性选择。
UCPOL 选择数据改变和采样发生在上升沿或下降沿。</td></tr>
<tr><td>UCPOL</td><td>发送数据的改变</td><td>接收数据的采样</td></tr>
<tr><td>0</td><td>XCK 的上升沿</td><td>XCK 的下降沿</td></tr>
<tr><td>1</td><td>XCK 的下降沿</td><td>XCK 的上升沿</td></tr>
</table>

UBRRL – USPI 波特率寄存器低字节

UBRRL – USPI 波特率寄存器低字节								
地址: 0xC4					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UBRR7	UBRR6	UBRR5	UBRR4	UBRR3	UBRR2	UBRR1	UBRR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	UBRR[7:0]	USPI 波特率寄存器的低字节部分。 USPI 波特率寄存器包含 UBRRL 和 UBRRH 两部分，结合在一起用来设置通信的波特率。

UBRRH – USPI 波特率寄存器高字节

UBRRH – USPI 波特率寄存器高字节								
地址: 0xC5					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	-	-	-	-	UBRR11	UBRR10	UBRR9	UBRR8
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	描述
7:4	-	USPI 下保留。
3:0	UBRR[11:8]	USPI 波特率寄存器的高字节部分。 USPI 波特率寄存器包含 UBRRL 和 UBRRH 两部分，结合在一起用来设置通信的波特率。 UBRR = {UBRR[11:8], UBRRL}

		工作模式	波特率计算公式
		从机模式	波特率由外部主机决定
		主机模式	BAUD = f_{sys}/(2*(UBRR+1))

UDR – USPI 数据寄存器

UDR – USPI 数据寄存器								
地址: 0xC6					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	UDR7	UDR6	UDR5	UDR4	UDR3	UDR2	UDR1	UDR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	UDR	USPI 发送和接收的数据。 USPI 发送数据缓冲器和接收数据缓冲器共享 USPI 数据寄存器 UDR。将数据写入 UDR 即写入发送数据缓冲器，从 UDR 读取数据即读取接收数据缓冲器。 在 5 到 8 位数据帧模式下，未使用的第 9 位被发送器忽略，而接收器则将它们设置为 0。 只有当 UCSRA 寄存器的 UDRE 标志为“1”时才能对发送缓冲器进行写操作，否则发送器的操作会出错。当发送移位寄存器为空时，发送器会把发送缓冲器中的数据加载到发送移位寄存器中，然后数据串行地从 TxD 引脚输出。 接收缓冲器包含一个两级 FIFO，一旦接收缓冲器被读取，FIFO 就会改变它的状态。

TWI – 双线串行总线(I2C)

- 简单且强大而灵活的通讯接口，只需要 2 线
- 支持主机和从机操作
- 器件可以工作于发送器模式或接收器模式
- 7 位地址空间允许有 128 个从机
- 支持多主机仲裁
- 高达 400Kbps 的数据传输率
- 完全可编程的从机地址以及公共地址
- 睡眠模式下地址匹配时可以唤醒

TWI 总线介绍

两线串行接口 TWI 很适合于典型的处理器应用。TWI 协议允许系统设计者只用两根双向的传输线就可以将 128 个不同的设备互连到一起。这两根线是时钟 SCL 和数据 SDA。外部硬件只需要在每根线上接两个上拉电阻。所有连接到总线上的设备都有自己的地址。TWI 协议解决了总线仲裁的问题。

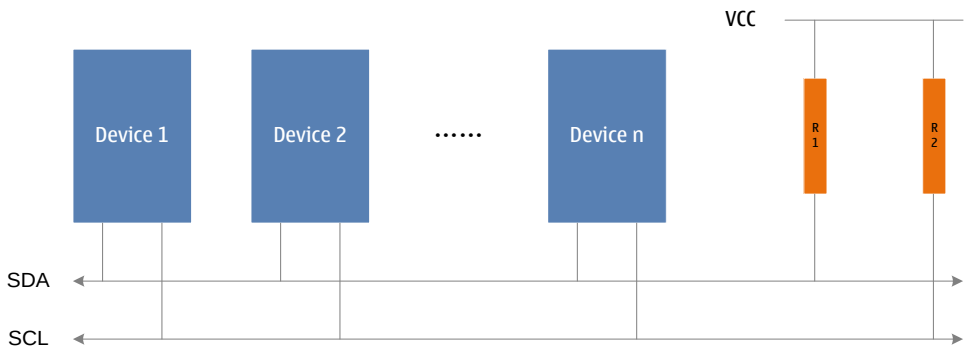
TWI 术语

以下定义的术语将在本节频繁出现。

术语	描述
主机	启动和停止传输的设备。主机还要负责产生 SCL 时钟。
从机	被主机寻址的设备
发送器	将数据放到总线上的设备
接收器	从总线上接收数据的设备

电气连接

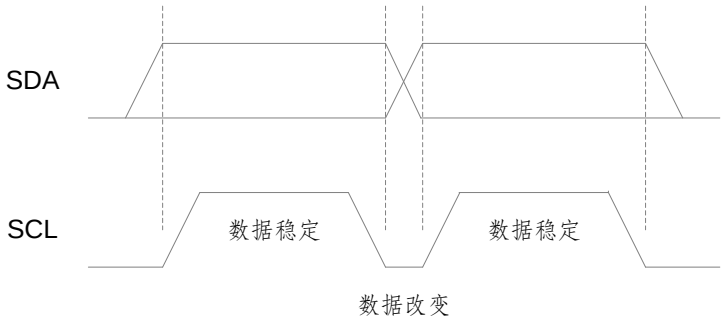
如下图所示，TWI 接口的两根线都通过上拉电阻与正电源连接。所有 TWI 兼容器件的总线驱动都是漏极开路或集电极开路的，这样就实现了对接口操作的线与功能。当 TWI 器件输出为“0”时，TWI 总线会产生低电平。当所有的 TWI 器件输出为三态时，总线允许上拉电阻将电压拉高。为保证所有的总线操作，凡是与 TWI 总线连接的器件都必须上电。



TWI 总线互连图

数据传输和帧结构

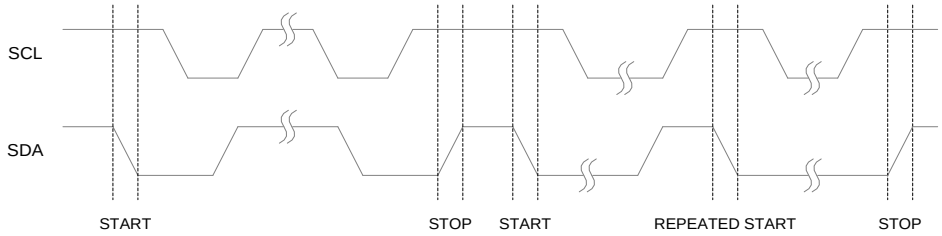
TWI 总线上的每一位数据传输都是和时钟同步的。当时钟线为高时，数据线上的电平必须保持稳定，除非是为了产生开始或停止状态。



TWI 数据有效性图

开始和停止状态

TWI 的传输由主机来启动和停止。主机在总线上发出 START 状态以其送数据传输，发出 STOP 状态以停止数据传输。在 START 和 STOP 状态之间，总线被认为是忙碌的，不允许其它主机试图占用总线的控制权。有一种特殊情况只允许发生在 START 和 STOP 状态之间产生一个新的 START 状态，这被称为 REPEATED START 状态，适用于当前主机在不放弃总线控制的情况下启动新的传输。REPEATED START 之后直到下一个 STOP 之前，总线仍然被认为是忙碌的。这与 START 是一致的，因此在本文档中，如果没有特殊说明，均采用 START 来表述 START 和 REPEATED START。如下图所示，START 和 STOP 条件是在 SCL 线为高时，改变 SDA 线的电平状态。



START、REPEATED START 和 STOP 状态图

地址包格式

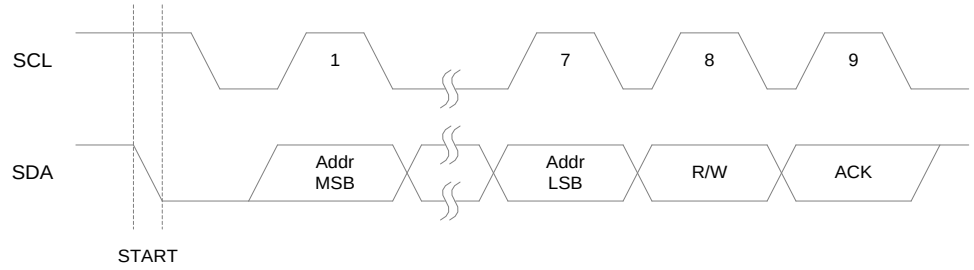
所有 TWI 总线上传输的地址包都是 9 位数据长度，由 7 位地址，1 位 READ/WRITE 控制位和 1 位应答位组成。当 READ/WRITE 位为“1”，则执行读操作；当 READ/WRITE 位为“0”时，执行写操作。从机被寻址后，必须在第 9 个 SCL（ACK）周期通过拉低 SDA 线做出应答。若该从机忙或有其它原因无法响应主机，则应在 ACK 周期保持 SDA 线为高。然后主机可以发出 STOP 状态或 REPEATED START 状态重新开始发送。

地址包包括一个从机地址和一个读或写控制位，分别用 SLA+R 或 SLA+W 来表示。

地址字节的 MSB 位首先发生。除了保留地址“00000000”被留用作广播呼叫以及所有形如“1111xxxx”格式的地址需要保留作将来使用外，其它从机地址可由设计者自由分配。

当发生广播呼叫时，所有的从机应在 ACK 周期通过拉低 SDA 线来做出应答。当主机需要发送相同的信息给多个从机时可以使用广播功能。当广播呼叫地址加上 WRITE 位被发送到总线上以后，所有需要响应该广播呼叫的从机将在 ACK 周期拉低 SDA 线。所有这些响应了广播呼叫的从机将会接收紧跟的数据包。需要注意的是，发送广播呼叫地址加上 READ 位是没有意义的，因为如果几个从机同时发送不同的数据会带来总线冲突。

地址包格式如下图所示：

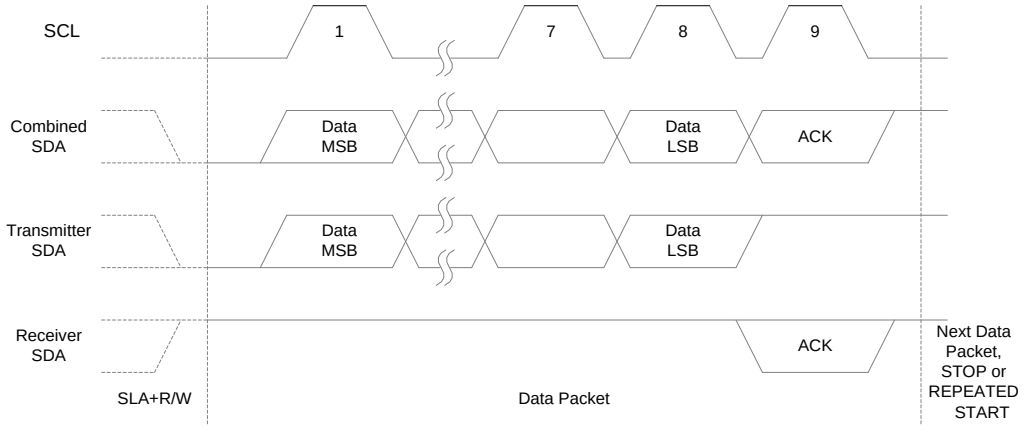


TWI 地址包格式图

数据包格式

所有 TWI 总线上传输的数据包都是 9 位数据长度，由 1 个数据字节和 1 位应答位组成。在数据传输期间，主机负责产生传输时钟 SCL 和 START 及 STOP 状态，发送器发送要传输的字节数据，接收器产生接收响应。确认信号 ACK 是接收器在第 9 个 SCL（ACK）周期通过拉低 SDA 线来产生的。如果接收器在 ACK 周期保持 SDA 线为高，则发出的是未确认信号 NACK。当接收器已经接收到了最后一个字节，或者由于某些原因不能再接收任何数据，则应该在收到最后字节后通过发送 NACK 来告知发送器。数据字节的 MSB 位先传输。

数据包格式如下图所示：

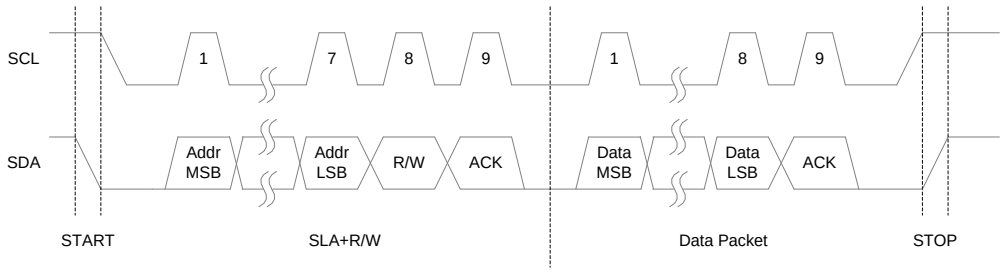


TWI 数据包格式图

组合地址和数据包的传输，一次传输基本上由 1 个 START，1 个 SLA+R/W，1 个或多个

数据包以及 1 个 STOP 组成。只有 START 和 STOP 的空信息是非法的。可以使用 SCL 线的线与功能来实现主机与从机的握手。从机可以通过拉低 SCL 线来延长 SCL 的地电平周期。当主机设定的时钟速度远远快于从机，或从机需要额外的时间来处理数据时，这个特性就非常有用。从机延长 SCL 的低电平周期并不会影响 SCL 的高电平周期，它仍然是由主机决定的。由此可知，从机可以通过改变 SCL 的占空比来降低 TWI 的数据传输速度。

下图所示的是一个典型的数据传输。注意 SLA+R/W 与 STOP 之间可以传送多个字节，取决于应用软件的实现协议。



典型的 TWI 传输

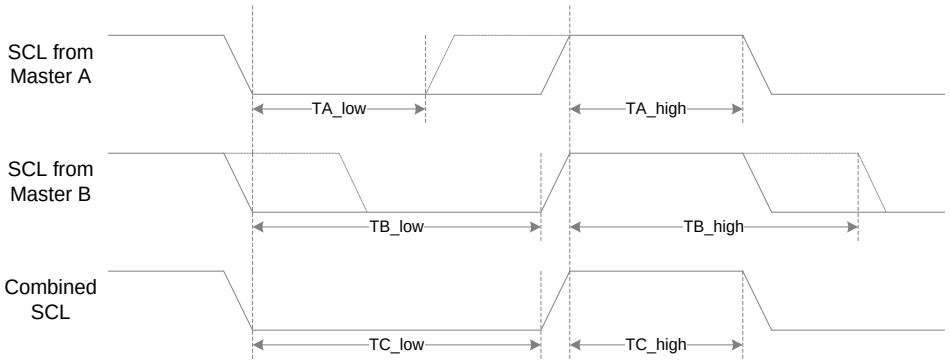
多主机系统及其仲裁和同步

TWI 协议允许总线上有多个主机，并采用了特殊的措施来保证即使两个或多个主机同时启动传输也能够像普通传输一样处理。多主机系统会出现两个问题：

1. 实现的算法只允许多主机中的一个主机完成传输。当其它主机发现它们失去选择权后必须停止它们的传输。这个选择的过程就叫做仲裁。当竞争中的主机发现其仲裁失败后，应立即切换到从机模式来检测自己是否被获得总线控制权的主机寻址。事实上多主机同时开始传输时不应该被从机检测到，即不允许破坏正在总线上传送的数据。
2. 不同的主机可能使用不同的 SCL 频率。为保证传送的一致性，必须设计一种同步主机串行时钟的方案。这会简化仲裁过程。

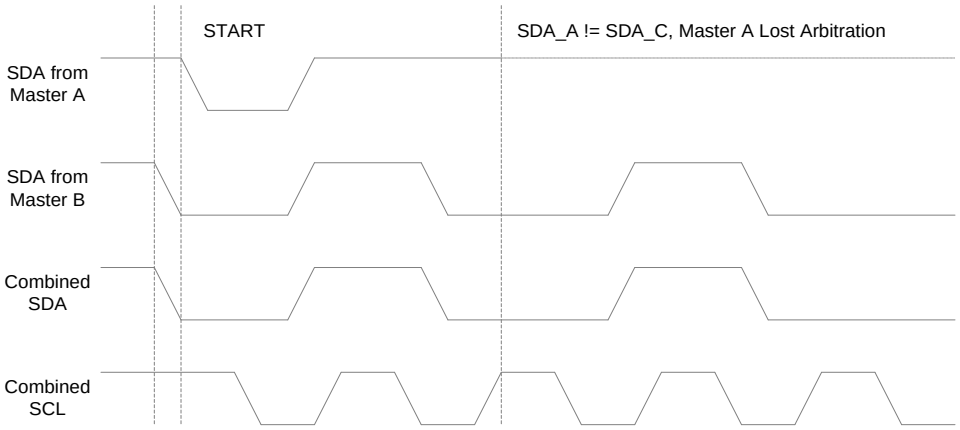
总线的线与功能就是用来解决上述问题的。所有主机的串行时钟都会线与到一起产生一个组合时钟，其高电平时间等于所有主机时钟中最短的一个，其低电平则等于所有主机时钟中最长的一个。所有主机都监听 SCL，当组合 SCL 时钟变高或变低时，它们可以有效地分别开始计算各自 SCL 高电平和低电平溢出周期。

多主机的 SCL 时钟同步机制如下图所示：



多主机 SCL 时钟同步时序图

输出数据之后所有的主机都持续监听 SDA 线来实现仲裁。如果从 SDA 读回的数值与主机输出的数值不匹配，该主机即失去仲裁。要注意的是，主机输出高电平的 SDA，而另一个主机输出低电平的 SDA 时才会失去仲裁。失去仲裁的主机应立即转换为从机模式，并检测是否被寻址。失去仲裁的主机必须将 SDA 线置高，但在当前的数据或地址包结束之前还可以产生时钟信号。仲裁将会持续到系统只剩下一个主机，这可能会占用多个比特。如果多个主机对相同的从机寻址，仲裁将会持续到数据包。



两个主机之间的仲裁

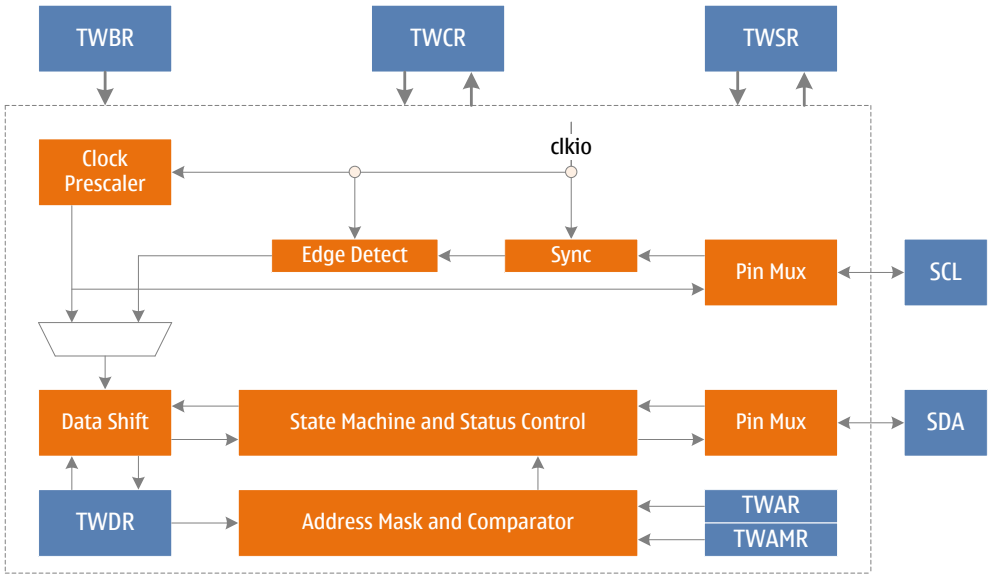
注意不允许在以下情形进行仲裁：

- 一个 REPEATED START 状态与一个数据位之间；
- 一个 STOP 状态与一个数据位之间；
- 一个 REPEATED START 状态与 STOP 状态之间；

应用软件必须考虑上述情况，保证不会出现这些非法仲裁情形。这意味着在多主机系统中，所有的数据传输必须由相同的 SLA+R/W 与数据包组成。换句话说，所有的传送必须包含相同数目的数据包，否则仲裁结果无法定义。

TWI 模块综述

TWI 模块的结构图如下图所示。



TWI Block 结构图

TWI 模块主要包括比特率发生器，总线接口单元，地址比较器和控制单元等。具体见下列详细描述。

比特率发生器单元

比特率发生器单元主要控制主机模式下的 SCL 时钟周期。SCL 时钟周期由 TWI 比特率寄存器 TWBR 和 TWI 状态寄存器 TWSR 中的预分频控制位共同决定。从机操作不受比特率或预分频设置的影响，但要保证从机的工作时钟至少是 SCL 频率的 16 倍。注意，从机可能会延长 SCL 的低电平周期，从而降低 TWI 总线的平均时钟频率。SCL 时钟频率有以下的计算公式产生：

$$f_{scl} = f_{sys}/(16 + 2*TWBR*4^{TWPS})$$

其中，TWBR 为 TWI 比特率寄存器的数值，TWPS 为 TWI 状态寄存器中的预分频控制位。

总线接口单元

总线接口单元包括数据和地址移位寄存器 TWDR，START/STOP 控制器和仲裁判定硬件电路。

TWDR 包含要发送的地址或数据字节，或者已接收的地址或数据字节。除了包含 8 位的 TWDR，总线接口单元还包括发送或接收的 ACK/NACK 寄存器。这个 ACK/NACK 寄存器不能直接被应用软件访问。当接收数据时，它可以通过 TWI 控制寄存器 TWCR 来置位或清零。当发送数据时，接收到的 ACK/NACK 值由 TWI 状态寄存器 TWSR 中的 TWS 值来反映。

START/STOP 控制器负责产生和检测 START，REPEATED START 和 STOP 状态。当 MCU 处于某些休眠模式时，START/STOP 控制器仍可以检测 START 和 STOP 状态，当被 TWI 总线上的主机寻址时将 MCU 从休眠模式唤醒。

如果 TWI 以主机模式启动了数据传输，仲裁检测电路将持续监听总线，以确定是否仍拥有总线控制权。当 TWI 模块丢失总线控制权后，控制单元将会执行正确的动作并产生合适的状态码来通知 MCU。

地址匹配单元

地址匹配单元用来检查接收到的地址字节是否与 TWI 地址寄存器中的 7 位地址相匹配。当 TWAR 寄存器中的 TWI 广播呼叫识别使能位（TWGCE）置位，从总线接收到的地址也会与广播地址比较。一旦地址匹配成功，控制单元将执行正确的动作。TWI 模块可以响应或不响应主机的寻址，这取决于 TWCR 寄存器的设置。即使在休眠模式下，地址匹配单元也可以比较地址，若被总线上的主机寻址，则将 MCU 从休眠模式唤醒。

控制单元

控制单元负责监听总线并根据 TWCR 的设置产生相应的响应。当 TWI 总线上发生需要应用软件参与的事件时，TWI 中断标志位 TWINT 将会被置位。在接下来的一个时钟周期，TWI 状态寄存器 TWSR 将会被更新为表明该事件的状态码。在 TWINT 被置位时，TWSR 包含确切的状态信息。在其它时间里，TWSR 为一个特殊的状态码，表示没有确切的状态信息。一旦 TWINT 标志位被置位，SCL 线就一直保持低电平，暂停总线上的 TWI 传输，让应用软件处理事件。

下列情形下，TWINT 标志位将置位：

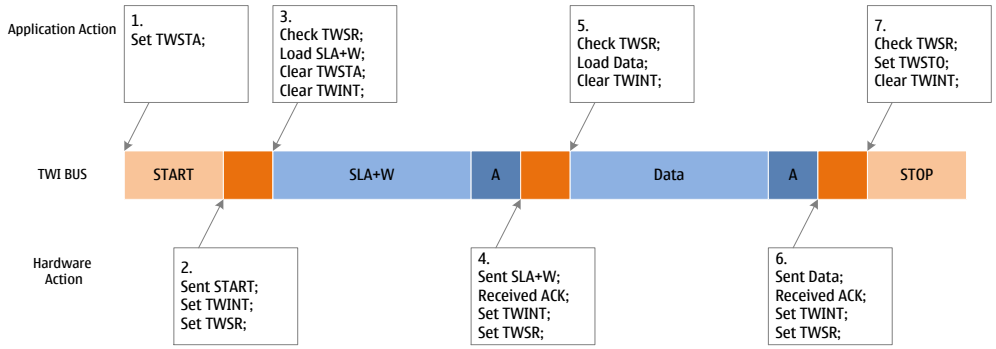
- TWI 传送完 START/REPEATED START 状态后
- TWI 传送完 SLA+R/W 后
- TWI 传送完一个地址字节后
- TWI 总线仲裁失败后
- TWI 被主机寻址后（从机地址匹配或广播方式）
- 被寻址作为从机工作时，收到 STOP 或 REPEATED START 后
- 由非法的 START 或 STOP 状态所引起的总线错误时

TWI 的使用

TWI 接口是面向字节和基于中断的。所有的总线事件，如接收到一个字节或发送了一个 START 信号等，都会产生一个 TWI 中断。由于 TWI 是基于中断的，因此在 TWI 字节传送的过程中，应用软件可以自如的进行其它操作。TWCR 寄存器中的 TWI 中断使能位 TWIE 和全局中断使能位一起来控制在 TWINT 标志位置位时是否产生 TWI 中断。如果 TWIE 位被清零，应用软件必须采用查询 TWINT 标志位的方式来检测 TWI 总线上的动作。

当 TWINT 标志位被置位时，表示 TWI 接口完成了当前的操作，等待应用软件的响应。在这种情况下，TWI 状态寄存器 TWSR 中包含了反映当前总线状态的状态码。应用软件可以通过设置 TWCR 和 TWDR 寄存器，来决定在接下来的 TWI 总线周期 TWI 接口该如何工作。

下图给出的是应用程序与 TWI 接口连接的例子。该例中，主机期望发送一个字节的数据给从机。这里的描述很简单，接下来的章节会有更详细的展示。



TWI 典型的传输过程图

图中所示的 TWI 传输过程为：

1. TWI 传输的第一步是发送 START。通过往 TWCR 寄存器写入特定值，指示 TWI 硬件发送 START 信号。写入的值将在随后详细说明。在写入的值中要置位 TWINT，这非常重要，往 TWINT 位写“1”会清零该位。TWCR 寄存器的 TWINT 置位期间 TWI 不会启动任何操作。一旦软件清零 TWINT 位，TWI 模块立即启动 START 信号的传送。
2. 当 START 状态发送完毕，TWCR 的 TWINT 标志位会被置位，TWSR 更新为的新的状态码，表示 START 信号成功发送。
3. 应用程序查看 TWSR 的值，确定 START 状态已经成功发送。如果 TWSR 显示为其它值，应用程序可以执行一些特殊操作，比如调用错误处理程序。当确定状态码与预期一致后，程序将 SLA+W 的值载入到 TWDR 寄存器中。TWDR 寄存器可同时在地址和数据中使用。随后软件往 TWCR 寄存器写入特定值，指示 TWI 硬件发送 TWDR 中的 SLA+W 的值。写入的值将在随后详细说明。在写入的值中要置位 TWINT，来清零 TWINT 标志位。TWCR 寄存器的 TWINT 置位期间 TWI 不会启动任何操作。一旦软件清零 TWINT 位，TWI 模块立即启动地址包的传送。
4. 当地址包发送完毕后，TWCR 的 TWINT 标志位会被置位，TWSR 更新为新的状态码，表示地址包成功发送。状态码同样会反映从机是否响应该地址包。
5. 应用程序查看 TWSR 的值，确定地址包已成功发送，收到的 ACK 为期望值。如果 TWSR 显示为其它值，应用程序可以执行一些特殊操作，比如调用错误处理程序。当确定状态码与预期一致后，程序将 Data 的值载入到 TWDR 寄存器中。随后软件往 TWCR 寄存器写入特定值，指示 TWI 硬件发送 TWDR 中的 Data 的值。写入的值将在随后详细说明。在写入的值中要置位 TWINT，来清零 TWINT 标志位。TWCR 寄存器的 TWINT 置位期间 TWI 不会启动任何操作。一旦软件清零 TWINT 位，TWI 模块立即启动数据包的传送。
6. 当数据包发送完毕后，TWCR 的 TWINT 标志位会被置位，TWSR 更新为新的状态码，表示数据包成功发送。状态码同样会反映从机是否响应该数据包。
7. 应用程序查看 TWSR 的值，确定数据包已成功发送，收到的 ACK 为期望值。如果 TWSR 显示为其它值，应用程序可以执行一些特殊操作，比如调用错误处理程序。当确定状态码与预期一致后，软件往 TWCR 寄存器写入特定值，指示 TWI 硬件发送 STOP 信号。写入的值将在随后详细说明。在写入的值中要置位 TWINT，来清零 TWINT 标志位。TWCR 寄存器的 TWINT 置位期间 TWI 不会启动任何操作。一旦软件清零 TWINT 位，TWI 模块立即启动 STOP 信号的传送。需要注意的是，在 STOP 信号发送完毕之后 TWINT 不会被置位。

尽管示例比较简单，但它包含了 TWI 数据传输过程中的所有规则。总结如下：

- 当 TWI 完成一次操作并等待应用程序的反馈时，TWINT 标志置位。SCL 时钟线会被一直

拉低直到 TWINT 被清零；

- 当 TWINT 标志置位，用户必须更新所有 TWI 寄存器的值为与下一个 TWI 总线周期相关的值。例如，TWDR 寄存器必须载入下一个总线周期要发送的值。
- 当更新完所有的寄存器，同时完成其它必要的操作之后，应用程序写 TWCR 寄存器。在写 TWCR 时，TWINT 位必须被置位，用来清零 TWINT 标志。TWINT 被清零之后，TWI 开始执行由 TWCR 设定的操作。

传输模式

TWI 可以工作在下面 4 种主要的模式：主机发送器（MT），主机接收器（MR），从机发送器（ST）和从机接收器（SR）。同一应用下可以使用多种模式。例如，TWI 可以使用 MT 模式往 TWI EEPROM 写入数据，用 MR 模式从 EEPROM 读取数据。如果该系统上还有其它主机，有些也可能往 TWI 发送数据，则会使用 SR 模式。这是由应用软件来决定采用何种模式。

下面会对这些模式进行详细说明。在每种模式下的数据传输中，会结合图片来描述可能的状态码。这些图片包含了如下的缩写：

S： Start 状态
Rs： REPEATED START 状态
R： 读操作标志位（SDA 为高电平）
W： 写操作标志位（SDA 为低电平）
A: 应答位（SDA 为低电平）
NA： 无应答位（SDA 为高电平）
Data：8 位数据字节
P： STOP 状态
SLA： 从机地址

图片中的圆圈用来表示 TWINT 标志置位，圆圈中的数字表示 TWSR 寄存器中的状态码，其中预分频控制位被屏蔽为“0”。在这些地方，应用程序必须执行相应的操作来继续或完成 TWI 传输。TWI 传输会被挂起，直到 TWINT 标志位被清零。

当 TWINT 标志被置位，TWSR 中的状态码用来决定适当的软件操作。各表格中给出了每个状态码下所需的软件操作和后续串行传输的细节。注意表格里 TWSR 中的预分频控制位被屏蔽为“0”。

主机发送模式

在主机发送模式中，TWI 会发送一定数量的数据字节到从机接收器。为了进入主机模式，必须发送 START 信号。接下来的地址包格式决定 TWI 是进入主机发送器模式还是主机接收器模式。如果发送 SLA+W，则进入主机发送模式。如果发送 SLA+R，则进入主机接收模式。这一章节所提到的状态码均假设预分频控制位为“0”。

通过往 TWCR 寄存器写入下列数值来发出 START 信号：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

TWEN 位必须置“1”来使能 TWI 接□，TWSTA 置“1”来发送 START 信号，TWINT 置“1”来清零

TWINT 标志位。TWI 模块检测总线状态，在总线空闲时立即发送 START 信号。当发送完 START 后，硬件置位 TWINT 标志位，同时更新 TWSR 的状态码为 0x08。

为了进入主机发送模式，必须发送 SLA+W。这可通过下面操作来完成。先往 TWDR 寄存器写入 SLA+W，然后往 TWINT 位写“1”清零 TWINT 标志位来继续传输，即往 TWCR 寄存器写入下列数值来发送 SLA+W：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

当 SLA+W 发送完成且收到应答信号后，TWINT 又被置位，同时 TWSR 的状态码更新。可能的状态码为 0x18、0x20 或 0x38。各个状态码下合适的响应会在状态码表格中详细描述。

当 SLA+W 发送成功后，可以开始发送数据包。这可通过往 TWDR 寄存器写入数据来完成。TWDR 只有在 TWINT 标志位为高时才可以写入。否则，访问被忽略，同时写冲突标志位 TWWC 会被置位。更新完 TWDR 后，往 TWINT 位写“1”清零 TWINT 标志位来继续传输。即往 TWCR 寄存器写入下列数值来发送数据：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

当数据包发送完成且收到应答信号后，TWINT 又被置位，同时 TWSR 的状态码更新。可能的状态码为 0x28 或 0x30。各个状态码下合适的响应会在状态码表格中详细描述。

当数据发送成功后，可以继续发送数据包。这个过程一直重复，直到最后一个字节发送完毕。主机产生 STOP 信号或 REPEATED START 信号整个传输才结束。

通过往 TWCR 寄存器写入下列数值来发出 STOP 信号：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	1	x	1	0	x

通过往 TWCR 寄存器写入下列数值来发出 REPEATED START 信号：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

在发送 REPEATED START（状态码为 0x10）之后，TWI 接口可以再次访问相同的从机，或访问新的从机而不用发送 STOP 信号。REPEATED START 使得主机可以在不丢失总线控制权的情况下在不同从机之间，主机发送器和主机接收器模式之间进行切换。

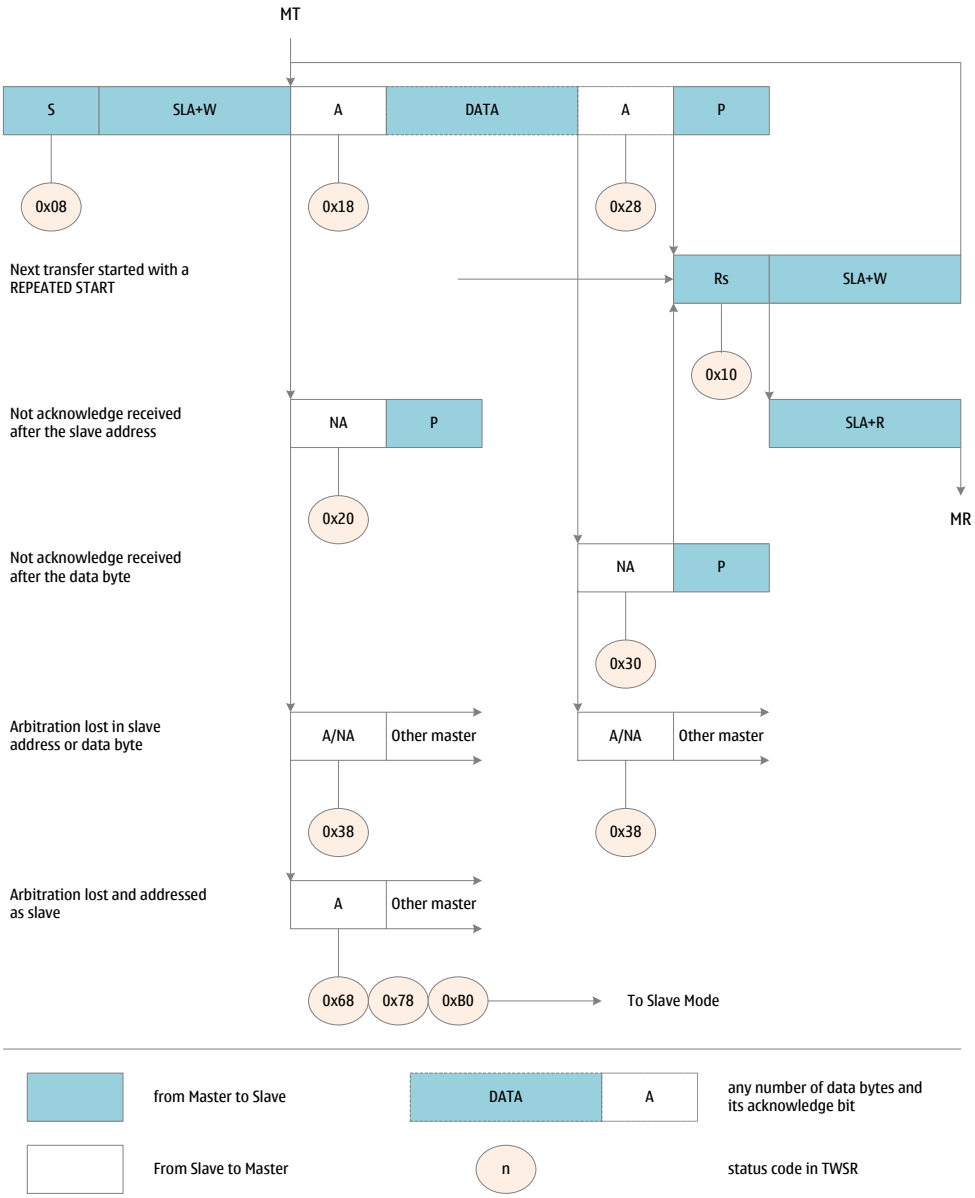
主机发送模式下的状态码及相应的操作如下表所示：

主机发送模式的状态码表

状态码	总线和硬件状态	应用软件的响应					硬件的下一步动作
		读/写 TWDR	对 TWCR 的操作				
			STA	STO	TWINT	TWEA	
0x08	START 已发送	加载 SLA+W	0	0	1	x	将发送 SLA+W； 将接收 ACK 或 NACK
0x10	REPEATED START 已发	加载 SLA+W	0	0	1	x	将发送 SLA+W； 将接收 ACK 或 NACK

	送	加载 SLA+R	0	0	1	x	将发送 SLA+R； 将接收 ACK 或 NACK; 将切换到 MR 模式
0x18	SLA+W 已发送； 接收到 ACK	加载数据	0	0	1	x	将发送数据； 将接收 ACK 或 NACK
		无操作	1	0	1	x	将发送 REPEATED START
		无操作	0	1	1	x	将发送 STOP； 将复位 TWSTO 标志
		无操作	1	1	1	x	将发送 STOP； 将复位 TWSTO 标志； 将发送 START
0x20	SLA+W 已发送； 接收到 NACK	加载数据	0	0	1	x	将发送数据； 将接收 ACK 或 NACK
		无操作	1	0	1	x	将发送 REPEATED START
		无操作	0	1	1	x	将发送 STOP； 将复位 TWSTO 标志
		无操作	1	1	1	x	将发送 STOP； 将复位 TWSTO 标志； 将发送 START
0x28	数据字节已发送；接收到 ACK	加载数据	0	0	1	x	将发送数据； 将接收 ACK 或 NACK
		无操作	1	0	1	x	将发送 REPEATED START
		无操作	0	1	1	x	将发送 STOP； 将复位 TWSTO 标志
		无操作	1	1	1	x	将发送 STOP； 将复位 TWSTO 标志； 将发送 START
0x30	数据字节已发送；接收到 NACK	加载数据	0	0	1	x	将发送数据； 将接收 ACK 或 NACK
		无操作	1	0	1	x	将发送 REPEATED START
		无操作	0	1	1	x	将发送 STOP； 将复位 TWSTO 标志
		无操作	1	1	1	x	将发送 STOP； 将复位 TWSTO 标志； 将发送 START
0x38	SLA+W 或数据仲裁失败	无操作	0	0	1	x	将释放总线； 将进入未寻址从机模式
		无操作	1	0	1	x	将在空闲时发送 START

主机发送模式的格式和状态如下图所示：



主机发送模式的格式和状态图

主机接收模式

在主机接收模式中，TWI 会从从机发送器接收一定数量的数据字节。为了进入主机模式，必须发送 START 信号。接下来的地址包格式决定 TWI 是进入主机发送器模式还是主机接收器模式。如果发送 SLA+W，则进入主机发送模式。如果发送 SLA+R，则进入主机接收模式。这一章节所提到的状态码均假设预分频控制位为“0”。

通过往 TWCR 寄存器写入下列数值来发出 START 信号：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

TWEN 位必须置“1”来使能 TWI 接口，TWSTA 置“1”来发送 START 信号，TWINT 置“1”来清零 TWINT 标志位。TWI 模块检测总线状态，在总线空闲时立即发送 START 信号。当发送完 START 后，硬件置位 TWINT 标志位，同时更新 TWSR 的状态码为 0x08。

为了进入主机接收模式，必须发送 SLA+R。这可通过下面操作来完成。先往 TWDR 寄存器写入 SLA+R，然后往 TWINT 位写“1”清零 TWINT 标志位来继续传输，即往 TWCR 寄存器写入下列数值来发送 SLA+R：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

当 SLA+R 发送完成且收到应答信号后，TWINT 又被置位，同时 TWSR 的状态码更新。可能的状态码为 0x38、0x40 或 0x48。各个状态码下合适的响应会在状态码表格中详细描述。

当 SLA+R 发送成功后，可以开始接收数据包。通过往 TWINT 位写“1”清零 TWINT 标志位来继续接收。即往 TWCR 寄存器写入下列数值来启动接收：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	0	x	1	0	x

当数据包接收完成且发送应答信号后，TWINT 又被置位，同时 TWSR 的状态码更新。可能的状态码为 0x50 或 0x58。各个状态码下合适的响应会在状态码表格中详细描述。

当数据接收成功后，可以继续接收数据包。这个过程一直重复，直到最后一个字节接收完毕。主机接收到最后一个字节后，必须发送 NACK 应答信号给从机发送器。主机产生 STOP 信号或 REPEATED START 信号整个接收才结束。

通过往 TWCR 寄存器写入下列数值来发出 STOP 信号：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	0	1	x	1	0	x

通过往 TWCR 寄存器写入下列数值来发出 REPEATED START 信号：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
1	x	1	0	x	1	0	x

在发送 REPEATED START（状态码为 0x10）之后，TWI 接口可以再次访问相同的主机，或访问新的主机而不用发送 STOP 信号。REPEATED START 使得主机可以在不丢失总线控制权的情况下在不同从机之间，主机发送器和主机接收器模式之间进行切换。

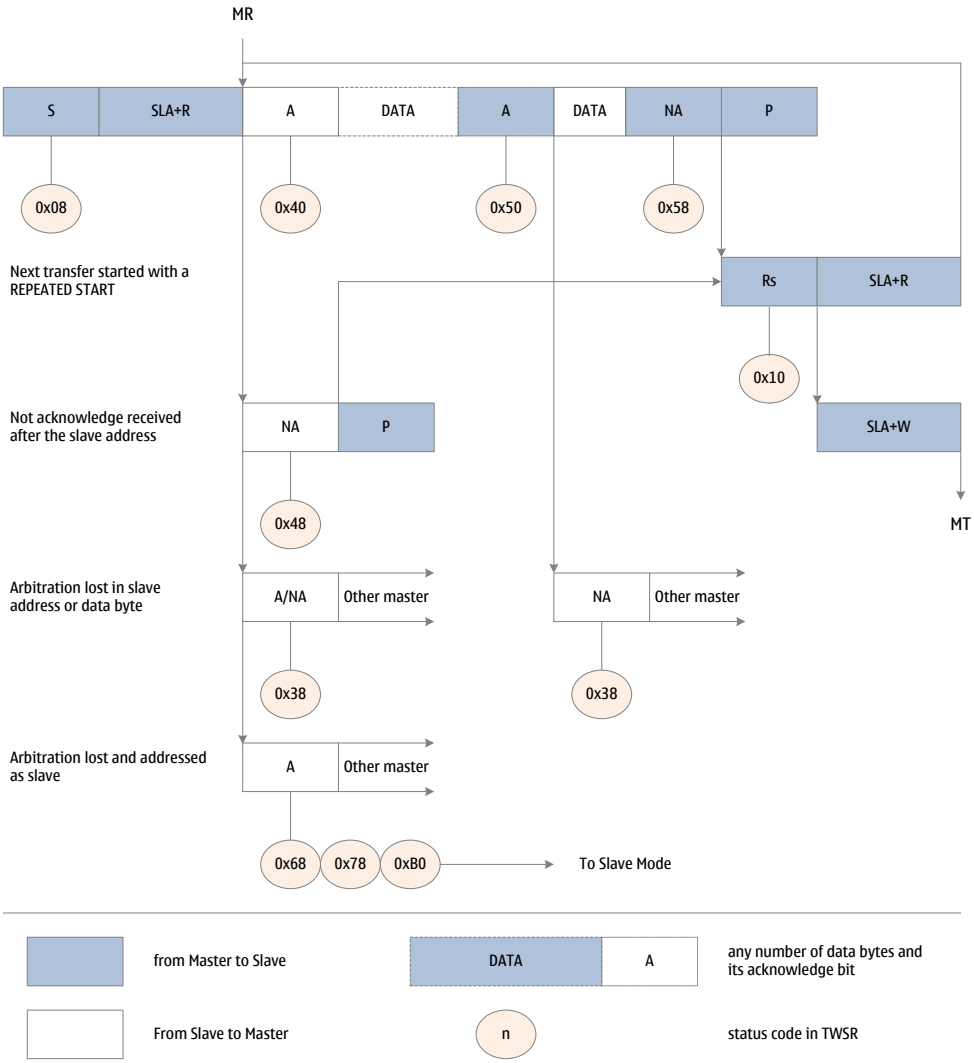
主机接收模式下的状态码及相应的操作如下表所示：

主机接收模式的状态码表

状态码	总线和硬件状态	应用软件的响应					硬件的下一步动作
		读/写 TWDR	对 TWCR 的操作				
			STA	STO	TWINT	TWEA	
0x08	START 已发送	加载 SLA+R	0	0	1	x	将发送 SLA+R； 将接收 ACK 或 NACK
0x10	REPEATED START 已	加载 SLA+R	0	0	1	x	将发送 SLA+R； 将接收 ACK 或 NACK

	发送	加载 SLA+W	0	0	1	x	将发送 SLA+W； 将接收 ACK 或 NACK; 将切换到 MT 模式
0x38	SLA+R 或数据仲裁失败	无操作	0	0	1	x	将释放总线； 将进入未寻址从机模式
		无操作	1	0	1	x	将在空闲时发送 START
0x40	SLA+R 已发送； 接收到 ACK	无操作	0	0	1	0	将接收数据； 将发送 NACK
		无操作	0	0	1	1	将接收数据； 将发送 ACK
0x48	SLA+R 已发送； 接收到 NACK	无操作	1	0	1	x	将发送 REPEATED START
		无操作	0	1	1	x	将发送 STOP； 将复位 TWSTO 标志
		无操作	1	1	1	x	将发送 STOP； 将复位 TWSTO 标志； 将发送 START
0x50	数据字节已接收； ACK 已发送	读取数据	0	0	1	0	将接收数据； 将发送 NACK
		读取数据	0	0	1	1	将接收数据； 将发送 ACK
0x58	数据字节已接收； NACK 已发送	读取数据	1	0	1	x	将发送 REPEATED START
		读取数据	0	1	1	x	将发送 STOP； 将复位 TWSTO 标志
		读取数据	1	1	1	x	将发送 STOP； 将复位 TWSTO 标志； 将发送 START

主机接收模式的格式和状态如下图所示：



主机接收模式的格式和状态图

从机接收模式

在从机接收模式中，可以从主机发送器接收一定数量的数据字节。这一章节所提到的状态码均假设预分频控制位为“0”。

为启动从机接收模式，要设置 TWAR 和 TWCR 寄存器。

TWAR 需设置如下：

TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
器件从机地址							

TWAR 的高 7 位是主机寻址时 TWI 接口会响应的从机地址。若 LSB 置位，TWI 会响应广播呼叫地址（0x00），否则忽略广播呼叫地址。

TWCR 需设置如下：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
0	1	0	0	0	1	0	x

TWEN 必须置位以使能 TWI 接口，TWEA 必须置位以使主机寻址（从机地址或广播呼叫）到自己时返回确认信息 ACK。TWSTA 和 TWSTO 必须清零。

初始化 TWAR 和 TWCR 之后，TWI 接口开始等待，直到自己的从机地址（或广播地址）被寻址。当紧跟着从机地址的数据方向位为“0”（表示写操作）时，TWI 进入从机接收模式。当数据方向位为“1”（表示读操作）时，TWI 进入从机发送模式。接收到自己的从机地址和写操作标志位后，TWINT 标志位被置位，有效的状态码也更新到 TWSR 中。各个状态码下合适的响应会在状态码表格中详细描述。需要注意的是，当主机模式下的 TWI 仲裁失败后也可以进入从机接收模式（见状态码 0x68 和 0x78）。

如果在传输过程中 TWEA 位被复位，TWI 将在接收到一个字节后返回 NACK（高电平）到 SDA 线上。这可用来表示从机不能接收更多的数据。当 TWEA 位为“0”时，TWI 也不会响应自己的从机地址。不过 TWI 仍会监听总线，一旦 TWEA 被置位，就可以恢复地址识别并响应。也就是说，可以利用 TWEA 暂时将 TWI 接口从总线中隔离出来。

在除空闲模式外的其它休眠模式时，TWI 接口的时钟可以被关闭。若是能了从机接收模式，接口将利用总线时钟继续响应从机地址或广播地址。地址匹配将唤醒 MCU。在唤醒期间，TWI 接口将保持 SCL 为低电平，直到 TWINT 标志被清零。当 TWI 接口时钟恢复正常后可以接收更多的数据。

从机接收模式的状态码如下表所示：

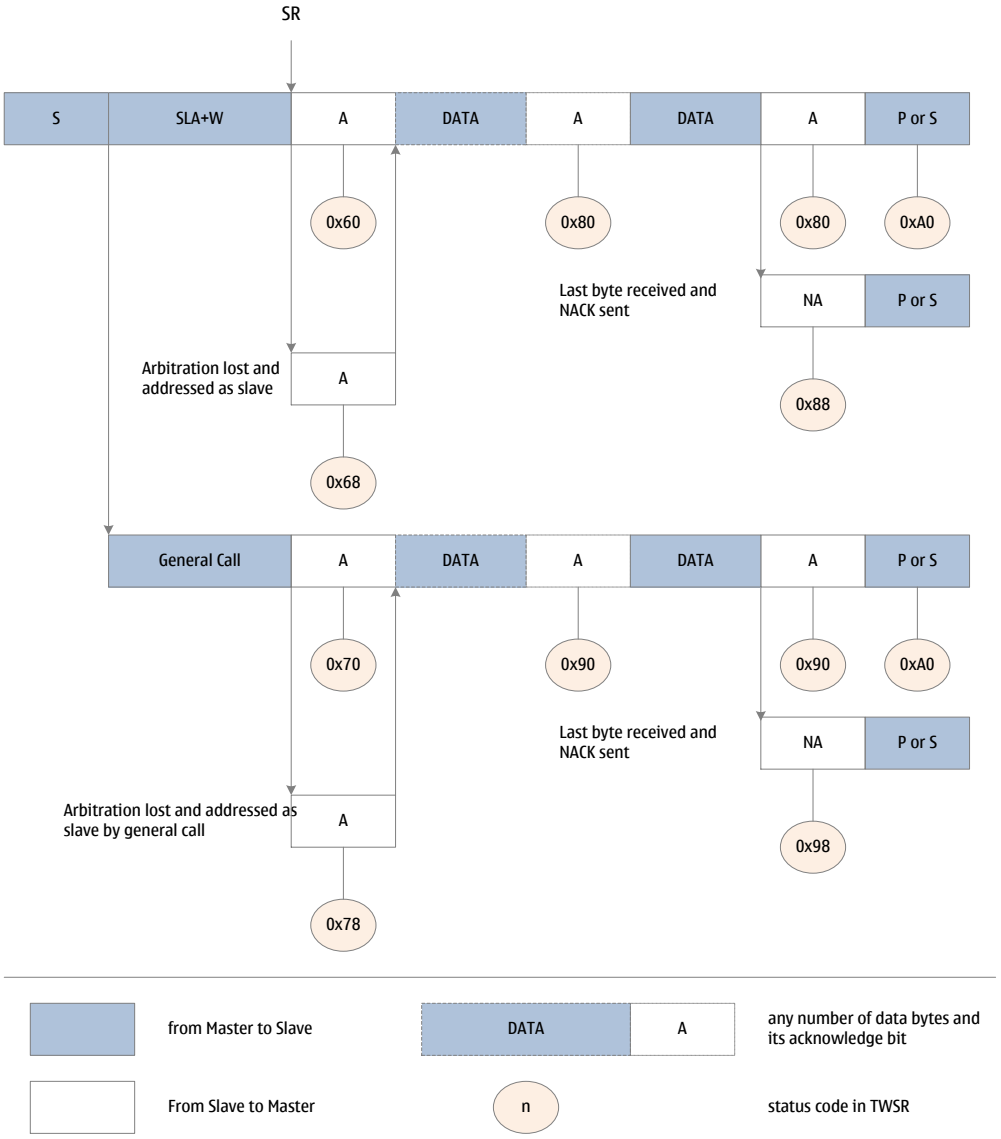
从机接收模式的状态码表

状态码	总线和硬件状态	应用软件的响应					硬件的下一步动作
		读/写 TWDR	对 TWCR 的操作				
			STA	STO	TWINT	TWEA	
0x60	SLA+W 已接收；ACK 已发送	无操作	x	0	1	0	将接收数据；将发送 NACK
		无操作	x	0	1	1	将接收数据；将发送 ACK
0x68	发送 SLA+R/W 时仲裁失败；SLA+W 已接收；ACK 已发送	无操作	x	0	1	0	将接收数据；将发送 NACK
		无操作	x	0	1	1	将接收数据；将发送 ACK
0x70	广播地址已接收；ACK 已发送	无操作	x	0	1	0	将接收数据；将发送 NACK
		无操作	x	0	1	1	将接收数据；将发送 ACK
0x78	发送 SLA+R/W 时仲裁失败；SLA+W 已接收；ACK 已发送	无操作	x	0	1	0	将接收数据；将发送 NACK
		无操作	x	0	1	1	将接收数据；将发送 ACK
0x80	自身数据已接收；ACK 已发送	读取数据	x	0	1	0	将接收数据；将发送 NACK
		读取数据	x	0	1	1	将接收数据；将发送 ACK
0x88	自身数据已接收；	读取	0	0	1	0	将切换到未寻址

	NACK 已发送	数据					从机模式； 将不响应从机地址和广播
		读取数据	0	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播
		读取数据	1	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播； 总线空闲时将发送 START
		读取数据	1	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播； 总线空闲时将发送 START
0x90	广播数据已接收； ACK 已发送	读取数据	x	0	1	0	将接收数据； 将发送 NACK
		读取数据	x	0	1	1	将接收数据； 将发送 ACK
0x98	广播数据已接收； NACK 已发送	读取数据	0	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播
		读取数据	0	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播
		读取数据	1	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播； 总线空闲时将发送 START
		读取	1	0	1	1	将切换到未寻址

		数据					从机模式； 将响应从机地址； TWGCE=1 时将响应广播； 总线空闲时将发送 START
0xA0	从机工作时接收到 STOP 或 REPEATED START	无操作	0	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播
		无操作	0	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播
		无操作	1	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播； 总线空闲时将发送 START
		无操作	1	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播； 总线空闲时将发送 START

从机接收模式的格式和状态图如下所示：



从机接收模式的格式和状态图

从机发送模式

在从机发送模式中，可以往主机接收器发送一定数量的数据字节。这一章节所提到的状态码均假设预分频控制位为“0”。

为启动从机接收模式，要设置 TWAR 和 TWCR 寄存器。

TWAR 需设置如下：

TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
器件从机地址							

TWAR 的高 7 位是主机寻址时 TWI 接口会响应的从机地址。若 LSB 置位，TWI 会响应广播呼叫地址（0x00），否则忽略广播呼叫地址。

TWCR 需设置如下：

TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
0	1	0	0	0	1	0	x

TWEN 必须置位以使能 TWI 接口，TWEA 必须置位以使主机寻址（从机地址或广播呼叫）到自己时返回确认信息 ACK。TWSTA 和 TWSTO 必须清零。

初始化 TWAR 和 TWCR 之后，TWI 接口开始等待，直到自己的从机地址（或广播地址）被寻址。当紧跟着从机地址的数据方向位为“0”（表示写操作）时，TWI 进入从机接收模式。当数据方向位为“1”（表示读操作）时，TWI 进入从机发送模式。接收到自己的从机地址和读操作标志位后，TWINT 标志位被置位，有效的状态码也更新到 TWSR 中。各个状态码下合适的响应会在状态码表格中详细描述。需要注意的是，当主机模式下的 TWI 仲裁失败后也可以进入从机发送模式（见状态码 0xB0）。

如果在传输过程中 TWEA 位被复位，TWI 将在发送最后一个字节后切换到未寻址从机模式。主机接收器为最后一个字节的传输给出 NACK 或 ACK 后，TWSR 寄存器中的状态码将会更新为 0xC0 或 0xC8。如果主机接收器继续传输操作，从机发送器不会响应，主机将会接收到全“1”的数据（即 0xFF）。当从机发送完最后一个字节的数据（TWEA 被清零）并期望得到 NACK 响应，而主机想要接收更多的数据而发送 ACK 作为响应时，TWSR 会更新为 0xC8。

当 TWEA 位为“0”时，TWI 也不会响应自己的从机地址。不过 TWI 仍会监听总线，一旦 TWEA 被置位，就可以恢复地址识别并响应。也就是说，可以利用 TWEA 暂时将 TWI 接口从总线中隔离出来。

在除空闲模式外的其它休眠模式时，TWI 接口的时钟可以被关闭。若是能了从机接收模式，接口将利用总线时钟继续响应从机地址或广播地址。地址匹配将唤醒 MCU。在唤醒期间，TWI 接口将保持 SCL 为低电平，直到 TWINT 标志被清零。当 TWI 接口时钟恢复正常后可以接收更多的数据。

从机发送模式的状态码如下表所示：

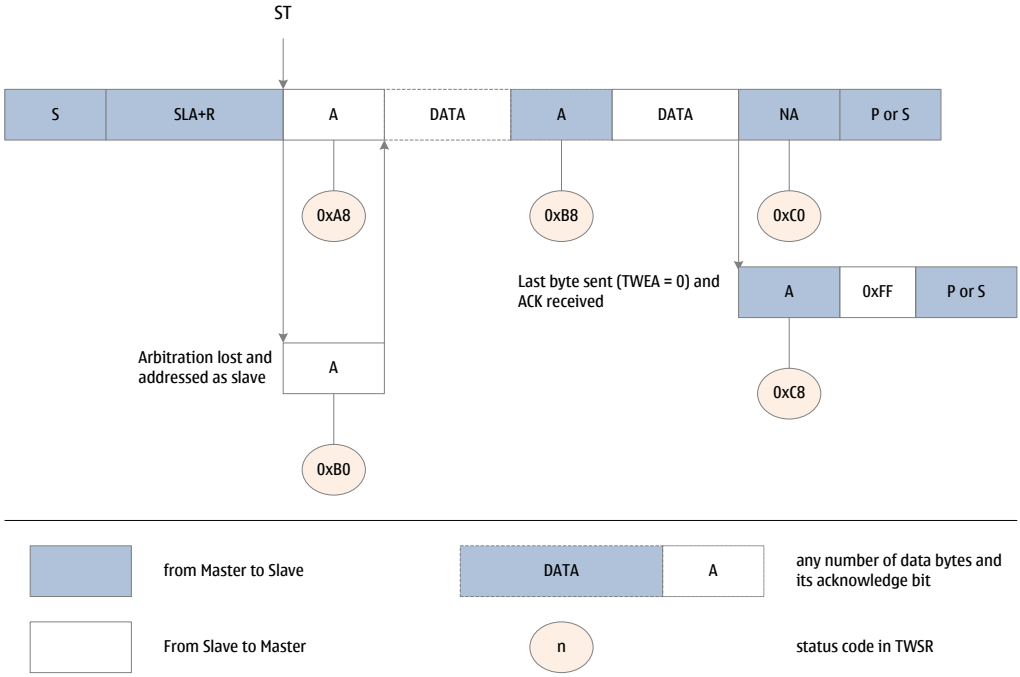
从机发送模式的状态码表

状态码	总线和硬件状态	应用软件的响应					硬件的下一步动作
		读/写 TWDR	对 TWCR 的操作				
			STA	STO	TWINT	TWEA	
0xA8	SLA+R 已接收；ACK 已发送	加载数据	x	0	1	0	将发送最后一个数据；期望接收 NACK
		加载数据	x	0	1	1	将发送数据；将接收 ACK
0xB0	发送 SLA+R/W 时仲裁失败；SLA+R 已接收；ACK 已发送	加载数据	x	0	1	0	将发送最后一个数据；期望接收 NACK
		加载数据	x	0	1	1	将发送数据；将接收 ACK

0xB8	数据已发送；ACK 已接收	加载数据	x	0	1	0	将发送最后一个数据； 期望接收 NACK
		加载数据	x	0	1	1	将发送数据； 将接收 ACK
0xC0	数据已发送； NACK 已接收	无操作	0	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播
		无操作	0	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播
		无操作	1	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播； 总线空闲时将发送 START
		无操作	1	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播； 总线空闲时将发送 START
0xC8	最后一个数据已发送； ACK 已接收	无操作	0	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播
		无操作	0	0	1	1	将切换到未寻址从机模式； 将响应从机地址； TWGCE=1 时将响应广播
		无操作	1	0	1	0	将切换到未寻址从机模式； 将不响应从机地址和广播； 总线空闲时将发送 START
		无操作	1	0	1	1	将切换到未寻址从机模式；

							将响应从机地址； TWGCE=1 时将响应广播； 总线空闲时将发送 START

从机发送模式的格式和状态如下图所示：



从机发送模式的格式和状态图

其他状态

有两个状态码没有相应的 TWI 状态定义，如下表所示：

其他状态码表

状态码	总线和硬件状态	应用软件的响应					硬件的下一步动作
		读/写 TWDR	对 TWCR 的操作				
			STA	STO	TWINT	TWEA	
0xF8	无状态信息； TWINT= 0	无操作	不操作 TWCR				等待或进行当前操作
0x00	非法的 START 或 STOP 引起的总线错误	无操作	0	1	1	x	只影响内部硬件；不会发送 STOP 到总线上；总线释放并清零 TWSTO 位

状态码 0xF8 表示当前没有相关信息，因为 TWINT 标志为“0”。这种状态可能发生在 TWI 接口没有参与串行传输或当前传输还没有完成。

状态 0x00 表示串行传输过程中发生了总线错误。当非法的 START 或 STOP 出现时总线错误就会发生。比如说在地址和数据、地址和 ACK 之间出现了 START 或 STOP。总线错误将置位 TWINT。为了从错误中恢复，必须置位 TWSTO，并通过写“1”以清零 TWINT。这将使 TWI 接口进入未寻址从机模式而不会产生 STOP，以及释放 SCL 和 SDA，并清零 TWSTO 位。

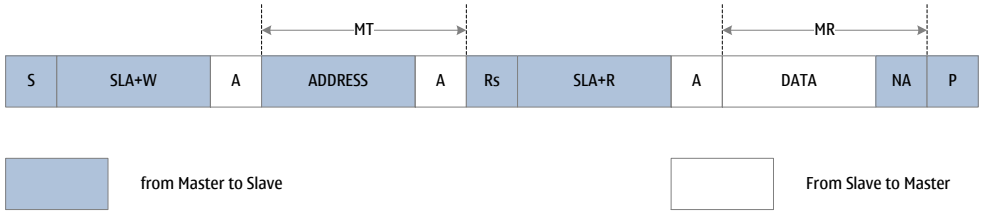
组合模式

在某些情况下，为了完成期望的工作，必须将几种 TWI 模式组合起来。例如，从串行 EEPROM 读取数据，典型的传输包括以下步骤：

1. 传输必须启动；
2. 必须告诉 EEPROM 应该读取数据的位置；
3. 必须完成读操作；
4. 传输必须结束。

注意数据可以从主机传送到从机，反之亦然。主机告诉从机要读取数据的位置，采用的是主机发送模式。接下来，从从机读取数据，采用的是主机接收模式。传输的方向会改变。主机必须保持各个阶段的总线控制权，所有的步骤是不间断的操作。如果在多主机系统中，在步骤 2 和 3 之间另有主机改变了读取数据的位置，则打破了这一原则，主机读取数据的位置会是错误的。改变数据传输的方向是通过在传送地址字节和接收数据之间发送 REPEATED START 来实现的。发送 REPEATED START 之后，主机仍拥有总线控制权。

下图描述了这个传输过程：



组合多种 TWI 模式来访问串行 EEPROM 图

多主机系统及仲裁

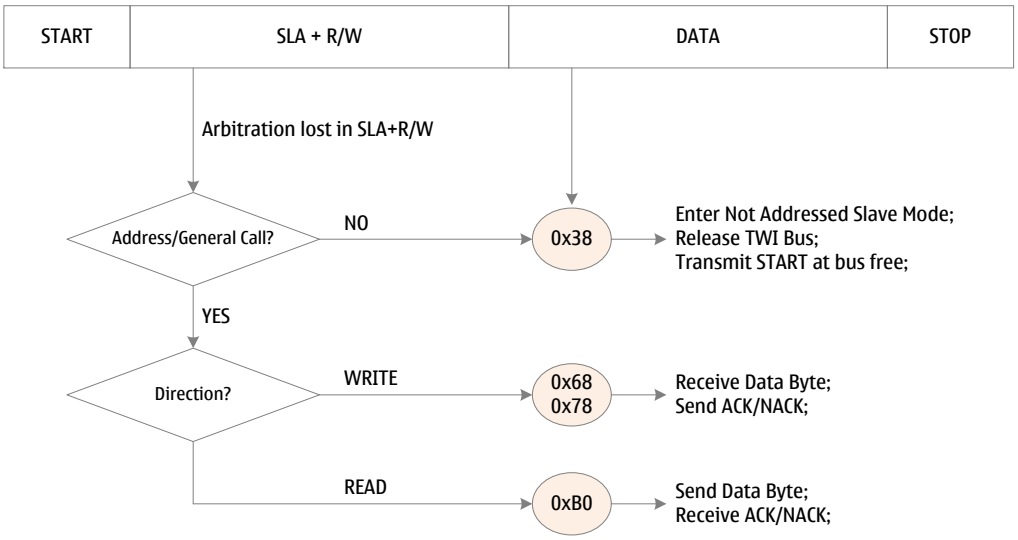
如果有多个主机连接在同一 TWI 总线上，它们中的一个或多个也许会同时开始数据传输。TWI 协议确保在这种情况下，通过一个仲裁过程，允许其中的一个主机进行传送也不会丢失数据。下面以两个主机试图向从机发送数据为例来描述总线仲裁的过程。

有几种不同的情况会产生总线仲裁过程：

- 两个或更多的主机同时与一个从机进行通信。在这种情况下，无论主机还是从机都不知道总线上有竞争；
- 两个或更多的主机同时对同一个从机进行不同的数据或操作方向访问。这种情况下就会发生仲裁，在 READ/WRITE 位或数据位。当有其它主机往 SDA 线上发送“0”时，往 SDA 线上发送“1”的主机就会仲裁失败。失败的主机将会切换到未被寻址的从机模式，或者等待总线空闲时发送一个新的 START 信号，这都取决于应用软件的操作。
- 两个或更多的主机访问不同的从机。在这种情况下，总线仲裁发生在 SLA 阶段。当有其它主机往 SDA 线上发送“0”时，往 SDA 线上发送“1”的主机就会仲裁失败。在 SLA 总线仲裁时失败的主机将切换到从机模式，并检查自己是否被获得总线控制权的主机寻址。如果被寻址，它将进入 SR 或 ST 模式，这取决于 SLA 后面的 READ/WRITE 位。如果未被寻

址，它将切换到未被寻址的从机模式，或者等待总线空闲时发送一个新的 START 信号，这取决于应用软件的操作。

下图描述了总线仲裁的过程：



总线仲裁过程图

寄存器定义

TWI 寄存器列表

寄存器	地址	默认值	描述
TWBR	0x B8	0x00	TWI 比特率寄存器
TWSR	0xB9	0x00	TWI 状态寄存器
TWAR	0xBA	0x00	TWI 地址寄存器
TWDR	0xBB	0x00	TWI 数据寄存器
TWCR	0xBC	0x00	TWI 控制寄存器
TWAMR	0xBD	0x00	TWI 地址屏蔽寄存器

TWBR – TWI 比特率寄存器

TWBR – TWI 比特率寄存器								
地址: 0xB8					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:0	TWBR[7:0]	TWI 比特率选择控制位。 TWBR 是比特率发生器分频因子。比特率发生器是一个分频器，用来在主机模式下产生 SCL 时钟。比特率的计算公式如下所示： $f_{scl} = f_{sys}/(16 + 2*TWBR*4^{TWPS})$。

TWSR – TWI 状态寄存器

TWSR – TWI 状态寄存器								
地址: 0xB9					默认值: 0xF8			
Bit	7	6	5	4	3	2	1	0
Name	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:3	TWS[7:3]	TWI 状态标志位。 5 位的 TWS 反应 TWI 逻辑和总线的状态。不同的状态值有不同的含义，具体见 TWI 工作模式的描述。从 TWSR 读到的值包括 5 位的状态值和 2 位的预分频控制位，在检测状态时应屏蔽预分频位为“0”。这是状态检测独立于预分频器的设置。
2	-	保留。
1	TWPS1	TWI 预分频控制高位。 TWPS1 和 TWPS0 一起组成 TWPS[1:0]，用来控制比特率预分频因子，和 TWBR 一起控制比特率。
0	TWPS0	TWI 预分频控制低位。 TWPS0 和 TWPS1 一起组成 TWPS[1:0]，用来控制比特率预分频因子，和 TWBR 一起控制比特率。

TWPS[1:0]	预分频因子
0	1
1	4
2	16
3	64

TWAR – TWI 地址寄存器

TWAR – TWI 地址寄存器								
地址: 0xBA					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	TWAR6	TWAR5	TWAR4	TWAR3	TWAR2	TWAR1	TWAR0	TWGCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7:1	TWA[6:0]	TWI 从机地址位。 TWA 为 TWI 从机地址。当 TWI 工作在从机模式下时，TWI 将根据这个地址进行响应。主机模式不需要此地址。但在多主机系统中，也需要设置从机地址以便其它主机访问。
0	TWGCE	TWI 广播识别使能控制位。 当设置 TWGCE 位为“1”时，使能 TWI 总线广播识别。 当设置 TWGCE 位为“0”时，禁止 TWI 总线广播识别。 当 TWGCE 置位且接收到的地址帧为 0x00 时，TWI 模块会响应此总线广播。

TWDR – TWI 数据寄存器

TWDR – TWI 数据寄存器								
地址: 0xBB					默认值: 0xFF			
Bit	7	6	5	4	3	2	1	0
Name	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:0	TWD[7:0]	TWI 数据寄存器。 TWD 是将要传送总线上的下一个字节，或者是刚从总线上接收到的上一个字节。						

TWCR – TWI 控制寄存器

TWCR – TWI 控制寄存器								
地址: 0xBC					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
R/W	R/W	R/W	R/W	R/W	R	R/W	-	R/W
Bit	Name	描述						
7	TWINT	TWI 中断标志位。 当 TWI 完成当前工作，希望应用软件介入时，硬件将置位 TWINT 位。若全局中断置位且 TWIE 位置位时，将产生 TWI 中断，MCU 将执行 TWI 中断服务程序。当 TWINT 标志被置位时，SCL 信号的低电平将被延长。 TWINT 标志位只能通过往该位写“1”的方式来清零。即使执行中断服务程序，硬件也不会自动清零该位。同时要注意，清零该位将立即开启 TWI 的操作。因此，在清零 TWINT 位之前，要首先完成对 TWAR，TWAMR，TWSR 和 TWDR 寄存器的访问。							
6	TWEA	TWI 使能应答控制位。 TWEA 位控制应答脉冲的产生。当设置 TWEA 位为“1”，且满足如下条件之一时，将会在 TWI 总线上产生应答脉冲： 1） 收到器件的从机地址； 2） TWGCE 置位时收到广播呼叫； 3） 在主机接收或从机接收模式下收到一个字节的数据。 当设置 TWEA 位为“0”时，器件暂时和 TWI 总线脱离连接。置位后器件重新恢复地址识别。							
5	TWSTA	TWI 起始状态控制位。 当 CPU 希望自己成为 TWI 总线上的主机时需要置位 TWSTA 位。硬件将检测总线是否可用，当总线是空闲时，就在总线上产生起始状态。当总线非空闲时，TWI 将一直等到检测到停止状态出现，然后产生起始状态来声明自己希望成为主机。发送完起始状态之后软件必须清零 TWSTA 位。							

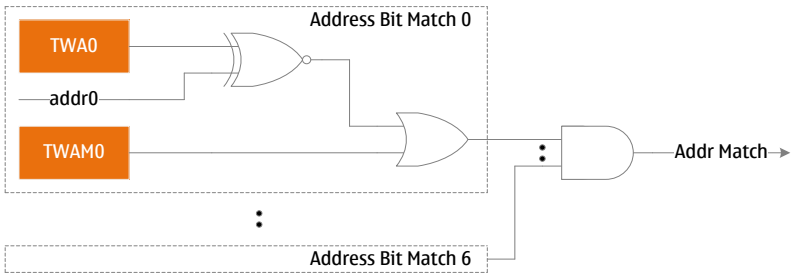
4	TWSTO	TWI 停止状态控制位。 在主机模式下当 TWSTO 位为“1”时，TWI 将在总线上产生停止状态，然后自动清零 TWSTO 位。在从机模式下，置位 TWSTO 位可以使 TWI 从错误状态恢复过来。这时不会产生停止状态，只会让 TWI 返回到一个定义好的未被寻址的从机模式，同时释放 SCL 和 SDA 信号线至高阻状态。
3	TWWC	TWI 写冲突标志位。 当 TWINT 标志位为低时，写 TWDR 寄存器将会置位 TWWC 标志位。 当 TWINT 标志位为高时，写 TWDR 寄存器将会清零 TWWC 标志位。
2	TWEN	TWI 使能控制位。 TWEN 位使能 TWI 操作并激活 TWI 接口。当设置 TWEN 位为“1”时，TWI 控制 IO 引脚连接到 SCL 和 SDA 引脚。当设置 TWEN 位为“0”时，TWI 接口模块被关闭，所有的传输被终止，包括正在进行的操作。
1	-	保留。
0	TWIE	TWI 中断使能控制位。 当设置 TWIE 位为“1”，且全局中断置位时，只要 TWINT 标志位为高，就会激活 TWI 中断请求。

TWAMR – TWI 地址屏蔽寄存器

TWAMR – TWI 地址屏蔽寄存器								
地址: 0xBD						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	TWAR6	TWAR5	TWAR4	TWAR3	TWAR2	TWAR1	TWAR0	TWGCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	Name	描述						
7:1	TWAM[6:0]	TWI 地址屏蔽控制位。 TWAM 为 7 位 TWI 从机地址屏蔽控制。 TWAM 的每一位用来屏蔽（禁止）TWAR 中相应地址位。当屏蔽位置位时，地址匹配逻辑将忽略接收到的地址位与 TWA 相应位的比较结果。下图给出了地址匹配逻辑的详细信息。						
0	-	保留。						

TWI 地址匹配逻辑

下图为 TWI 地址匹配逻辑框图：



模拟比较器 0 (AC0)

- 10mV 的比较精度
- 出厂失调校准
- 支持 3 路片外模拟输入
- 支持 ADC 的多路复用输入(ADMUX)
- 支持内部差分放大器输入(DFF0)
- 支持内部 8 位 DAC 输入(DA0)
- 可编程输出数字滤波控制

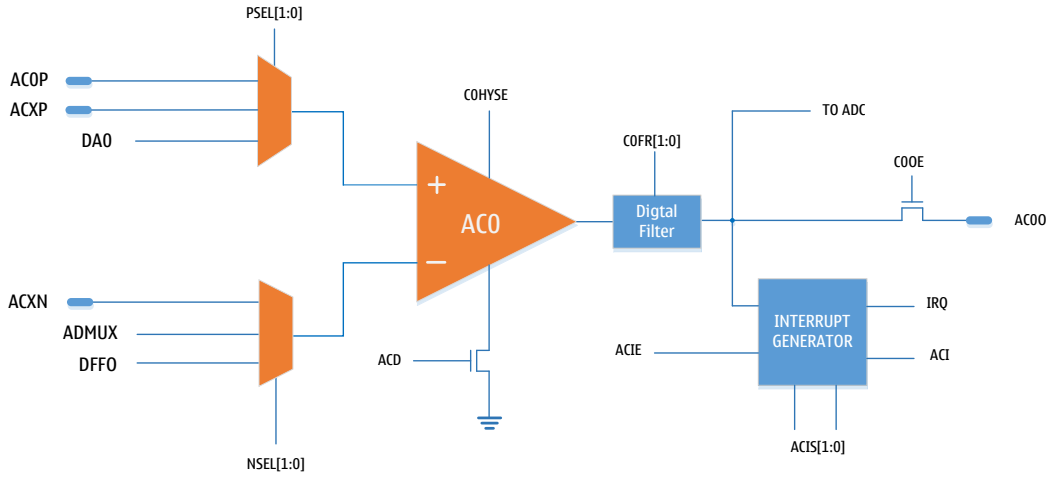
综述

模拟比较器对输入比较器正端与负极的电平进行比较，当正端电压比负端电压高时，模拟比较器的输出 AC0 被置位。当 AC0 的电平发生变化时，信号的边沿可用来触发中断。输出信号 AC0 还可用来触发定时计数器 1 的输入捕捉以及对定时器产生的 PWM 输出进行控制。

LGT8FX8P 集成模拟比较器 AC0，包括一个多路模拟输入选择器，比较器正、负端输入源可以选择来自外部端口或者来自多种内部产生的参考源。模拟比较器本身支持失调校准，可以保证比较器工作的一致性。比较器支持一个可选的硬件迟滞功能，用于改善比较器输出的稳定性。同时比较器输出端集成一个硬件可以编程数字滤波器，可以根据应用需求，选择合适的滤波设置，以获得更加稳定的比较输出。

比较器输出状态可以直接通过寄存器读取，也可以产生中断请求，实现更高效的实时事件俘获功能。比较器的输出也可以直接输出到外部 IO 端口。

运放/模拟比较器 0 的结构图如下图所示。



模拟比较器 0 功能示意图

模拟比较器的输入

模拟比较器的两个输入端都支持多种可选输入源。正端的输入三路可选：

1. 外部独立模拟输入 AC0P
2. 模拟比较器 0/1 公用模拟输入 ACXP
3. 内部 8 位 DAC 的输出 DA0

输入源的选择由控制状态寄存器 COSR 中的 COBG 位以及 COXR 寄存器的 COPSO 位共同控制，具体请参考本章节寄存器描述部分。

ACOP 为 ACO 专用正端模式输入通道。注意在不同封装片 ACOP 的脚位略有区别。QFP48 封装的 ACOP 为独立端口。QFP32 封装此 ACOP 端口与 PD6 并联到一个端口上。

ACXP 为比较器 0/1 公用正端输入。LGT8FX8P 内部有两个模拟比较器，ACXP 同时连接到两个比较器的正端多路复用选择器，便于实现两个比较器的协同工作。

DAO 来自内部 8 位 DAC 的输出。DAC 的参考源可以选择来自系统电源，内部参考或者来自外部参考的输入。DAC 的配置请参考 DAC 相关章节。

COBG	COPSO	ACO 正端输入源
0	0	ACOP
0	1	ACXP
1	0	DAO
1	1	关闭比较器正端输入通道

负端输入也可以选择三种不同的模拟输入:

1. 比较器 0/1 公用模拟输入 ACXN
2. ADC 多路器的输出 ADMUX
3. 内部差分放大器输出 DFFO

比较器负端输入通道选择由来自 ADC 模块的 ADCSRB 寄存器中的 CME00/01 位控制。当比较器负端输入选择为 ADMUX 时，需要通过 ADC 模块的 ADMUX 寄存器 CHMUX 位选择模拟输入通道，这种模式下，比较器的输入可以实现更加灵活的扩展。

ACXN 为比较器 0/1 公用的负端输入，便于实现比较器 0/1 的协同工作；

DFFO 来自内部的差分放大器输出。差分放大器可选 x1/x8/x16/x32 增益控制，可实现小信号的检测与测量。

CME01	CME00	ACO 负端输入源
0	0	ACXN
0	1	ADMUX
1	0	DFFO
1	1	关闭比较器负端输入通道

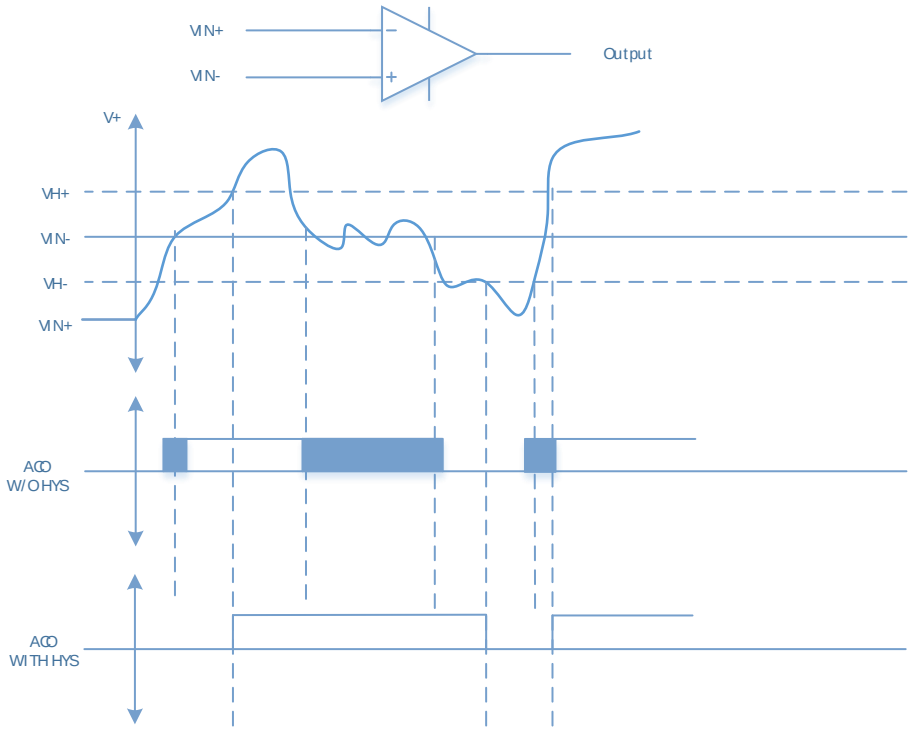
比较器输出滤波

比较器输出端内部支持一个可控的迟滞电。用户可以通过 COXR 寄存器的 COHYSE 位使能迟滞电路。迟滞电路可以消除比较器状态变化过程的不稳定状态，达到输出滤波功能。

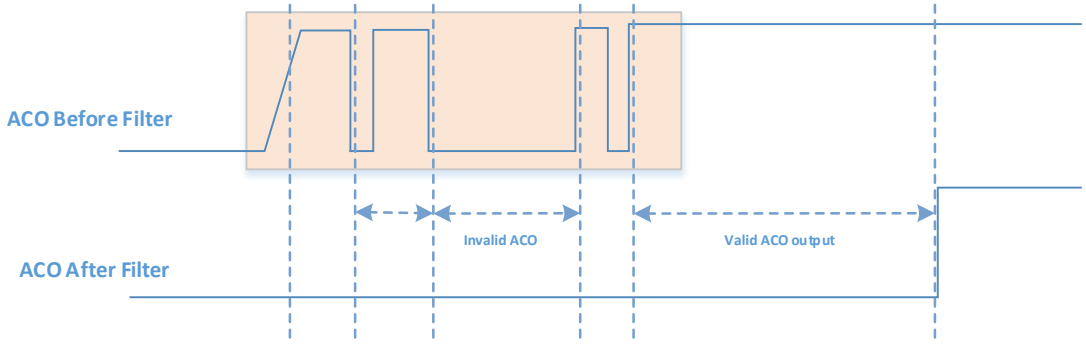
建议用户在使用比较器时，打开迟滞电路，获得一个稳定的比较器输出。

如下图所示，迟滞电路位于比较器模拟输出与数字输出之间。当比较器正端的输入电压 V_{IN+}大于(V_{IN-} + V_{H+})时，比较器 COUT 输出为高；当 V_{IN+}电压小于(V_{IN-} - V_{H-})时，比较器输出低。迟滞电路避免了当比较器正端电压接近负端电压时，电路本身带来的抖动。

比较器迟滞电压与比较器输出关系图:



尽管迟滞电路对于抑制接近比较器阀值的电压纹波非常有效，但实际应用环境中，输入信号会受到不同强度的干扰。较强的干扰可能会导致输入电平瞬间抬高，超出迟滞电路的阀值范围，无法被有效抑制。LGT8FX8P 在比较器输出端集成了一个可编程的数字滤波器，可以滤除瞬时干扰对比较器输出产生的影响。数字滤波器可以根据应用需求，选择合适的滤波时间宽度，只有当比较器的输出稳定持续满足滤波时间限制，滤波电路才更新比较器的输出。从而达到一个更加稳定的输出结果。



比较器输出滤波时序

ACO 的数字滤波通过 COXR 寄存器的 COFEN 以及 COFS 位控制，具体设置方式请参考本章寄存器定义部分。

比较器输出与 PWM 控制

LGT8FX8P 支持多通道 PWM 输出，PWM 信号可以与比较器模块配合使用。比较器的输出，可用于直接关断 PWM 信号，从而实现比较灵活的 PWM 保护方案。

与 PWM 输出相关的控制，请参考定时器章节的相关部分。

寄存器定义

COSR – ACO 控制和状态寄存器

COSR – ACO 控制和状态寄存器								
地址: 0x50					默认值: 0x80			
Bit	7	6	5	4	3	2	1	0
Name	COD	COBG	COO	COI	COIE	COIC	COIS1	COIS0
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	COD	模拟比较器禁止位。 当设置 COD 位为“1”时，模拟比较器被关闭。 当设置 COD 位为“0”时，模拟比较器被开启。
6	COBG	模拟比较器 0 正端输入源选择位。COBG 与 COXR 寄存器的 COPS0 位共同设置 ACO 正端输入源, {COBG, COPS0} = 00 = ACOP 作为正端输入 01 = ACXP 作为正端输入 10 = 内部 DAC 的输出作为正端输入 11 = 关闭 ACO 的正端输入源
5	COO	模拟比较器的输出状态位。 模拟比较器的输出经过同步之后直接连到 COO 位。软件可读取 COO 位的值来获取模拟比较器的输出值。
4	COI	模拟比较器的中断标志位。 当模拟比较器的输出事件触发了由 COIS 位定义的中断模式时，COI 位被置位。当中断使能位 COIE 为“1”且全局中断置位时，中断产生。执行模拟比较器中断服务程序时，COI 将自动清零，或对 COI 位写“1”也可清零该位。
3	COIE	模拟比较器的中断使能位。 当设置 COIE 位为 1，且使能全局中断，ACO 的中断被使能。 当设置 COIE 位为 0，ACO 的中断被禁止。
2	COIC	模拟比较器输入捕捉使能位 COIC = 1，定时计数器 1 的输入捕捉源来自模拟比较器的输出。 COIC = 0，定时计数器 1 的输入捕捉源来自外部引脚 ICP1。
1	COIS1	模拟比较器中断模式控制高位。
0	COIS0	模拟比较器中断模式控制低位。COIS0 和 COIS1 一起组成 COIS[1:0]，用来控制模拟比较器的中断触发方式。

COIS[1:0]	中断模式
00	ACO 的上升沿或下降沿触发
01	保留。
10	ACO 的下降沿触发
11	ACO 的上升沿触发

ADCSRB – ADC 控制和状态寄存器 B

ADCSRB – ADC 控制和状态寄存器 B								
地址: 0x7B	默认值: 0x00							
Bit	7	6	5	4	3	2	1	0
Name	CME01	CME00	CME11	CME10	ACTS	ADTS2	ADTS1	ADTS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	CME01	AC0 负端输入选择, CME0 = {CME01, CME00} 00: 外部端口 ACXN 作为 AC0 负端输入 01: ADC 多路复用输出作为 AC0 负端输入 10: 差分放大器输出作为 AC0 负端输入 11: 关闭 AC0 的负端输入源
6	CME00	
5	CME11	AC1 负端输入选择, CME1 = {CME11, CME10} 00: 外部端口 ACXN 作为 AC1 负端输入 01: 外部端口 AC1N 作为 AC1 负端输入 10: ADC 内部 1/5 分压作为 AC1 负端输入 11: 差分运放的输出作为 AC1 负端输入
4	CME10	
3	ACHS	AC 触发源通道选择 0 – AC0 输出作为 ADC 自动转换触发源 1 – AC1 输出作为 ADC 自动转换触发源
2:0	ADTS	见 ADC 寄存器描述。

C0XR – AC0 辅助控制寄存器

C0XR – AC0 辅助控制寄存器								
地址: 0x51	默认值: 0x00							
Bit	7	6	5	4	3	2	1	0
Name	-	C0OE	C0HYSE	C0PS0	C0WKE	C0FEN	C0FS1	C0FS0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	-	保留
6	C0OE	AC0 比较器输出到外部端口的使能控制 C0OE = 1, AC0 的比较器输出到外部端口 PD2 C0OE = 0, 禁止比较器输出到外部端口
5	C0HYSE	AC0 输出迟滞功能使能控制。 1 = 使能输出迟滞 0 = 禁用输出迟滞
4	C0PS0	AC0 正端输入源选择低位。 C0PS0 与 C0BG 共同控制 AC0 的正端输入源，请参考 C0SR 寄存器定义
3	C0WKE	AC0 用于休眠唤醒的使能控制。 1 = 使能比较器输出的唤醒功能

		0 = 关闭比较器输出的唤醒功能
2	COFEN	比较器数字滤波使能控制。 1 = 使能数字滤波器 0 = 禁用数字滤波器
1:0	COFS[1:0]	比较器数字滤波宽度设置 00 = 关闭 01 = 32us 10 = 64us 11 = 96us

模拟比较器 1 (AC1)

- 10mV 的比较精度
- 出厂失调校准
- 支持 4 路片外模拟输入
- 支持内部 1/5 分压器输入(VDO)
- 支持内部差分放大器输入(DFFO)
- 支持内部 8 位 DAC 输入(DAO)
- 可编程输出滤波控制

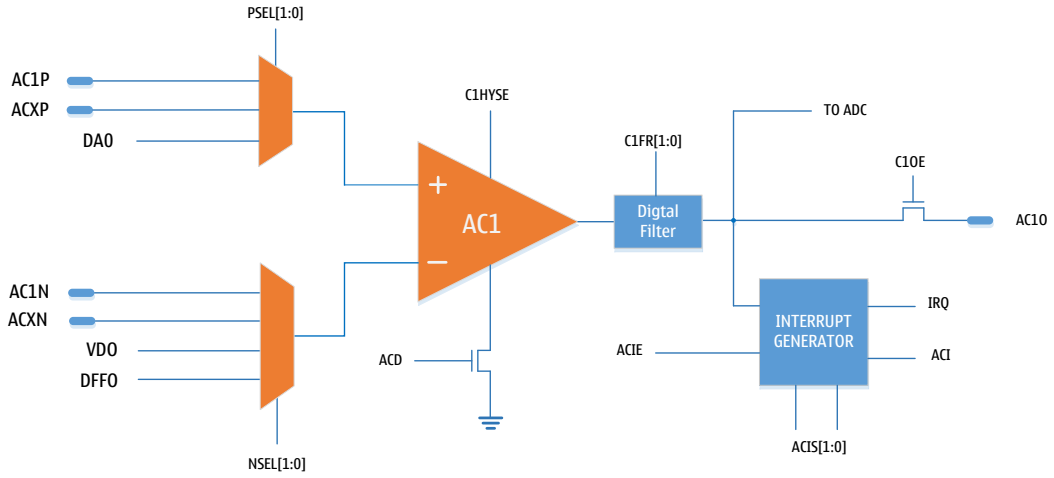
综述

模拟比较器对输入比较器正端与负极的电平进行比较，当正端电压比负端电压高时，模拟比较器的输出 ACO 被置位。当 ACO 的电平发生变化时，信号的边沿可用来触发中断。输出信号 ACO 还可用来触发定时计数器 1 的输入捕捉以及对定时器产生的 PWM 输出进行控制。

LGT8FX8P 集成模拟比较器 AC1，包括一个多路模拟输入选择器，比较器正、负端输入源可以选择来自外部端口或者来自多种内部产生的参考源。模拟比较器本身支持失调校准，可以保证比较器工作的一致性。比较器支持一个可选的硬件迟滞功能，用于改善比较器输出的稳定性。同时比较器输出端集成一个硬件可以编程数字滤波器，可以根据应用需求，选择合适的滤波设置，以获得更加稳定的比较输出。

比较器输出状态可以直接通过寄存器读取，也可以产生中断请求，实现更高效的实时事件俘获功能。比较器的输出也可以直接输出到外部 IO 端口。

模拟比较器 1 的结构图如下图所示。



模拟比较器 1 模块结构示意图

模拟比较器的输入

模拟比较器的两个输入端都支持多种可选输入源。正端的输入三路可选：

1. 外部独立模拟输入 AC1P
2. 模拟比较器 0/1 公用模拟输入 ACXP

3. 内部 8 位 DAC 的输出 DAO

输入源的选择由控制状态寄存器 C1SR 中的 C1BG 位以及 C1XR 寄存器的 C1PS0 位共同控制，具体请参考本章节寄存器描述部分。

AC1P 为 AC1 专用正端模式输入通道。

ACXP 为比较器 0/1 公用正端输入。LGT8FX8P 内部有两个模拟比较器，ACXP 同时连接到两个比较器的正端多路复用选择器，便于实现两个比较器的协同工作。

DAO 来自内部 8 位 DAC 的输出。DAC 的参考源可以选择来自系统电源，内部参考或者来自外部参考的输入。DAC 的配置请参考 DAC 相关章节。

C1BG	C1PS0	AC1 正端输入
0	0	AC1P
0	1	ACXP
1	0	DAO
1	1	关闭比较器正端输入通道

负端输入也可以选择 4 种不同的模拟输入:

1. 外部模拟输入 AC1N 作为 AC1 负端输入
2. 比较器 0/1 公用负端输入 ACXN
3. ADC 内部 1/5 分压器输出作为 AC1 的负端输入
4. 内部差分放大器输出 DFFO 作为 AC1 的负端输入

比较器负端输入通道选择由来自 ADC 模块的 ADCSRB 寄存器中的 CME11/10 位控制。当比较器负端输入选择为 ADC 内部多路分压器输出时，需要通过 ADC 模块的 ADCSRC 寄存器 VDS 位选择多路分压的输入参考源。

ACXN 为比较器 0/1 公用的负端输入，便于实现比较器 0/1 的协同工作；

DFFO 来自内部的差分放大器输出。差分放大器可选 x1/x8/x16/x32 增益控制，可实现小信号的检测与测量。

CME11	CME10	AC1 负端输入
0	0	ACXN
0	1	AC1N
1	0	VDO
1	1	DFFO

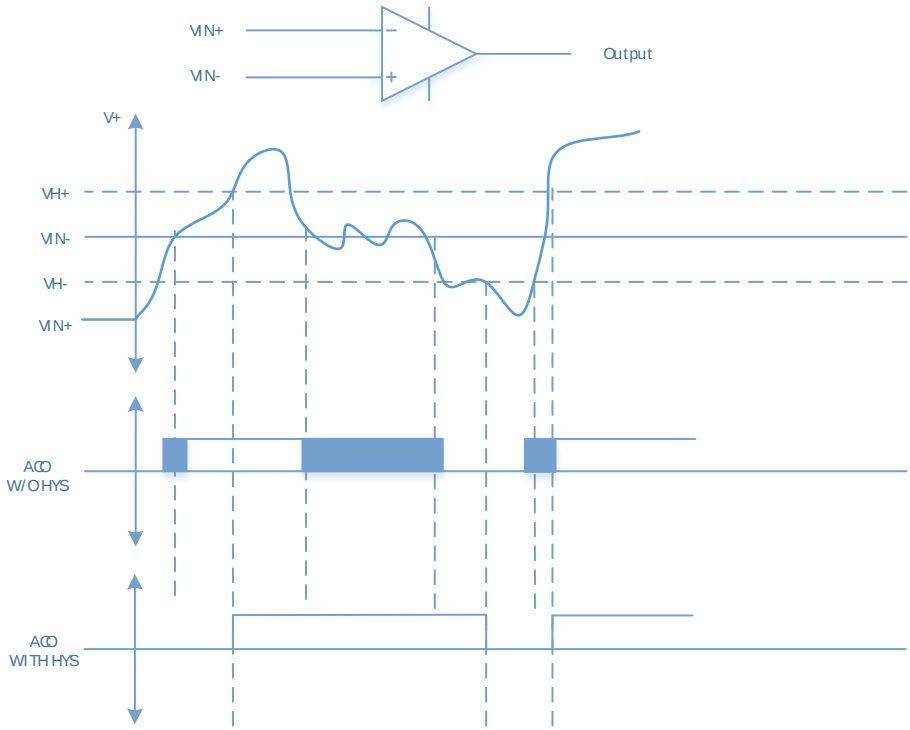
比较器输出滤波

比较器输出端内部支持一个可控的迟滞电。用户可以通过 C1XR 寄存器的 C1HYSE 位使能迟滞电路。迟滞电路可以消除比较器状态变化过程的不稳定状态，达到输出滤波功能。

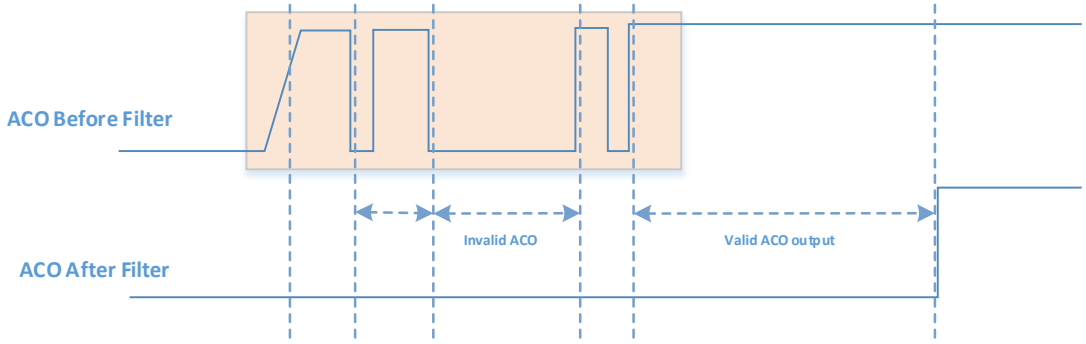
建议用户在使用比较器时，打开迟滞电路，获得一个稳定的比较器输出。

如下图所示，迟滞电路位于比较器模拟输出与数字输出之间。当比较器正端的输入电压 V_{IN+}大于(V_{IN-} + V_{H+})时，比较器 COUT 输出为高；当 V_{IN+}电压小于(V_{IN-} - V_{H-})时，比较器输出低。迟滞电路避免了当比较器正端电压接近负端电压时，电路本身带来的抖动。

比较器迟滞电压与比较器输出关系图:



尽管迟滞电路对于抑制接近比较器阀值的电压纹波非常有效，但实际应用环境中，输入信号会受到不同强度的干扰。较强的干扰可能会导致输入电平瞬间抬高，超出迟滞电路的阀值范围，无法被有效抑制。LGT8FX8P 在比较器输出端集成了一个可编程的数字滤波器，可以滤除瞬时干扰对比较器输出产生的影响。数字滤波器可以根据应用需求，选择合适的滤波时间宽度，只有当比较器的输出稳定持续满足滤波时间限制，滤波电路才更新比较器的输出。从而达到一个更加稳定的输出结果。



比较器输出滤波时序

AC1 的数字滤波通过 C1XR 寄存器的 C0FEN 以及 C1FS 位控制，具体设置方式请参考本章寄存器定义部分。

比较器输出与 PWM 控制

LGT8FX8P 支持多通道 PWM 输出，PWM 信号可以与比较器模块配合使用。比较器的输出，可用于直接关断 PWM 信号，从而实现比较灵活的 PWM 保护方案。

与 PWM 输出相关的控制，请参考定时器章节的相关部分。

寄存器定义

C1SR – AC1 控制和状态寄存器

C1SR – AC1 控制和状态寄存器								
地址: 0x2F					默认值: 0x80			
Bit	7	6	5	4	3	2	1	0
Name	C1D	C1BG	C1O	C1I	C1IE	C1IC	C1IS1	C1IS0
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	C1D	模拟比较器禁止位。 当设置 C1D 位为“1”时，模拟比较器被关闭。 当设置 C1D 位为“0”时，模拟比较器被开启。
6	C1BG	模拟比较器 1 正端输入源选择位。C1BG 与 C1XR 寄存器的 C1PS0 位共同设置 AC1 正端输入源, {C1BG, C1PS0} = 00 = AC1P 作为正端输入 01 = ACXP 作为正端输入 10 = 内部 DAC 的输出作为正端输入 11 = 关闭 AC1 的正端输入源
5	C1O	模拟比较器的输出状态位。 模拟比较器的输出经过同步之后直接连到 C1O 位。软件可读取 C1O 位的值来获取模拟比较器的输出值。
4	C1I	模拟比较器的中断标志位。 当模拟比较器的输出事件触发了由 C1IS 位定义的中断模式时，C1I 位被置位。当中断使能位 C1IE 为“1”且全局中断置位时，中断产生。执行模拟比较器中断服务程序时，C1I 将自动清零，或对 C1I 位写“1”也可清零该位。
3	C1IE	模拟比较器的中断使能位。 当设置 C1IE 位为 1，且使能全局中断，AC1 的中断被使能。 当设置 C1IE 位为 0，AC1 的中断被禁止。
2	C1IC	模拟比较器输入捕捉使能位 C1IC = 1，定时计数器 1 的输入捕捉源来自模拟比较器的输出。 C1IC = 0，定时计数器 1 的输入捕捉源来自外部引脚 ICP1。
1	C1IS1	模拟比较器中断模式控制高位。
0	C1IS0	模拟比较器中断模式控制低位。C1IS0 和 C1IS1 一起组成 C1PS[1:0]，用来控制模拟比较器的中断触发方式。

C1IS[1:0]	中断模式
00	AC1 的上升沿或下降沿触发
01	保留。
10	AC1 的下降沿触发
11	AC1 的上升沿触发

ADCSRB – ADC 控制和状态寄存器 B

ADCSRB – ADC 控制和状态寄存器 B								
地址: 0x7B	默认值: 0x00							
Bit	7	6	5	4	3	2	1	0
Name	CME01	CME00	CME11	CME10	ACTS	ADTS2	ADTS1	ADTS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	CME01	AC0 负端输入选择, CME0 = {CME01, CME00} 00: 外部端口 ACXN 作为 AC0 负端输入 01: ADC 多路复用输出作为 AC0 负端输入 10: 差分放大器输出作为 AC0 负端输入 11: 关闭 AC0 的负端输入源
6	CME00	
5	CME11	AC1 负端输入选择, CME1 = {CME11, CME10} 00: 外部端口 ACXN 作为 AC1 负端输入 01: 外部端口 AC1N 作为 AC1 负端输入 10: ADC 内部 1/5 分压作为 AC1 负端输入 11: 差分运放的输出作为 AC1 负端输入
4	CME10	
3	ACHS	AC 触发源通道选择 0 – AC0 输出作为 ADC 自动转换触发源 1 – AC1 输出作为 ADC 自动转换触发源
2:0	ADTS	见 ADC 寄存器描述。

C1XR – AC1 辅助控制寄存器

C1XR – AC1 辅助控制寄存器								
地址: 0x3A	默认值: 0x00							
Bit	7	6	5	4	3	2	1	0
Name	-	C1OE	C1HYSE	C1PS0	C1WKE	C1FEN	C1FS1	C1FS0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	-	保留
6	C1OE	AC1 比较器输出到外部端口的使能控制 C1OE = 1, AC1 的比较器输出到外部端口 PE5 C1OE = 0, 禁止比较器输出到外部端口
5	C1HYSE	AC1 输出迟滞功能使能控制。 1 = 使能输出迟滞 0 = 禁用输出迟滞
4	C1PS0	AC1 正端输入源选择低位。 C1PS0 与 C1BG 共同控制 AC1 的正端输入源，请参考 C1SR 寄存器定义
3	C1WKE	AC1 用于休眠唤醒的使能控制。 1 = 使能比较器输出的唤醒功能

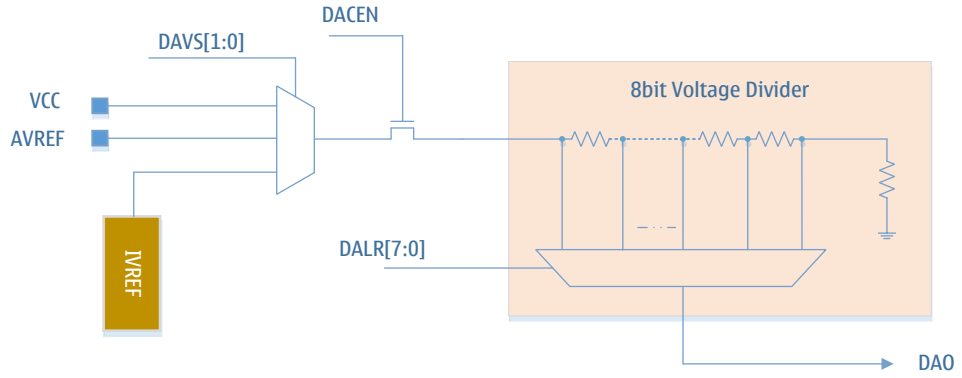
		0 = 关闭比较器输出的唤醒功能
2	C1FEN	比较器数字滤波使能控制。 1 = 使能数字滤波器 0 = 禁用数字滤波器
1:0	C1FS[1:0]	比较器数字滤波宽度设置 00 = 关闭 01 = 32us 10 = 64us 11 = 96us

数模转换器(DAC)

- 8 位数模转换输出
- DAC 输出可作为模拟比较器参考输入
- 支持 DAC 输出到外部端口(DAO)
- 可选 VCC/AVREF/IVREF 分压电源

综述

LGT8FX8P 内部集成一个 8 位可编程数模转换器(DAC)。DAC 的参考电源输入可以选择为来自系统工作电源，内部基准电压源或者来自芯片外部端口 AVREF 输入。DAC 的输出可选择作为内部比较器 AC0/1 的输入源，也可以直接输出至芯片的外部引脚上作为外部参考使用。当 DAC 输出至外部引脚时，不能直接用于驱动负载，需要通过电压跟随器或其他类似的驱动电路。DAC 内部结构如下图所示：



寄存器定义

DACON – DAC 控制寄存器

DACON– DAC 控制寄存器								
地址: 0xA0					0000_0000			
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	DACEN	DAOE	DAVS1	DAVS0
R/W	-	-	-	-	R/W	W/R	R/W	W/R
Bit	Name	描述						
7:4	-	保留						
3	DACEN	DAC 使能控制位 1 = 使能 DAC 模块 0 = 禁用 DAC 模块						
2	DAOE	DAC 输出到外部端口使能控制 1 = 使能 DAC 输出到外部端 PD4 0 = 禁止 DAC 输出到外部端口						
1	DAVS1	DAC 参考电压源选择位 1						
0	DAVS0	DAC 参考电压源选择位 0。[DVS1, DVS0] =						

		00：电压源选择系统工作电压 VCC 01：电压源选择为外部输入 AVREF 10：电压源选择为内部参考电压 11：关闭 DAC 参考源，同时也会关闭 DAC 模块

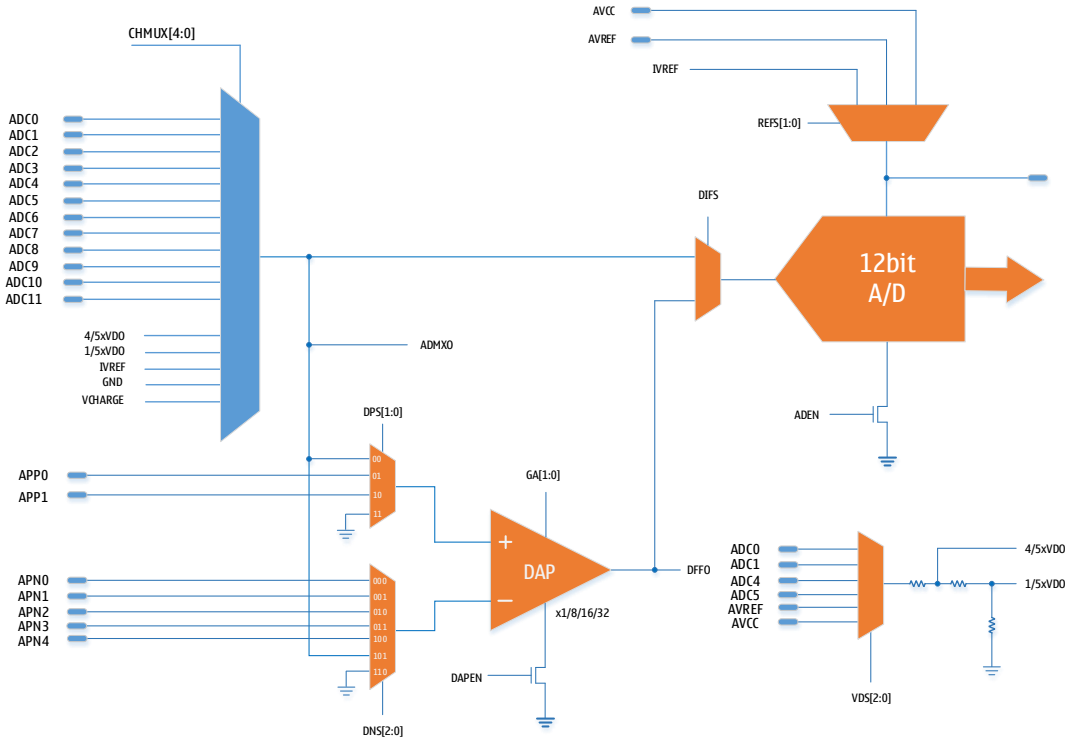
DALR – DAC 数据寄存器

VRCON1– DAC1 控制寄存器								
地址: 0xA1					0000_0000			
Bit	7	6	5	4	3	2	1	0
	DALR[7:0]							
R/W	W/R							
Bit	Name	描述						
7:0	DALR	DAC 数据寄存器，设置 DAC 模式输出电压大小 DAC 输出电压与 DALR 的关系： $V_{DAO} = V_{REF}*(DALR + 1)/256$ 其中： V_{DAO} 为 DAC 输出模拟电压 V_{REF} 为 DAC 参考电压源，由 DACON 寄存器的 DAVS 位选择						

12 位模数转换器(ADC)

- 12 位分辨率，DNL 为±1LSB，INL 为±1.5LSB
- 最高分辨率时采样率高达 500KSPS
- 12 路复用的单端输入通道
- 多路输入可编程增益差分放大器通道
- 输入电压范围为 0-VCC
- 内部 1.024V/2.048V/4.096V 参考电压
- 支持 AVCC 以及外部参考电压输入
- 内部多输入 1/5、4/5 分压电路
- 支持正负方向的失调校准
- 基于中断源的自动开始转换触发模式
- 支持上/下溢出的自动通道监测
- 转换结果支持可选对齐模式
- 转换结束中断请求

概述



ADC 结构图

模数转换器为一个 12 位的逐次逼近型 ADC。ADC 与一个 17 通道的模拟多路复用器连接，能对来自芯片外部端口 12 路模拟输入以及 5 通道内部电压源进行采样转换。ADC 内部集成一个可编程增益为 x1/x8/x16/x32 的差分运算放大器，放大器输入可来自外部端口或者 ADC 多路复用器的输出。差分运放的结果可作为 ADC 的模拟输入。

ADC 的内部模拟输入源包括来自 ADC 内部的多路输入分压器；内部参考电压源；内部模拟参考地以及来自触摸按键模块的模拟输出。内部多路输入分压器同时输出 4/5，1/5 两路

电压；分压器的输入可以选择来自外部端口的电平或者来自系统电源。

ADC 支持失调校准。失调校准的流程由软件控制。失调校准包括正、反两个方向的校准量。失调校准使能后，ADC 控制器将会自动使用正反两个校准值对 ADC 采样结果进行校准。

失调校准的方法请参考本章节相关部分。

ADC 的操作

ADC 通过逐次逼近的方法将输入的模拟电压转换成一个 12 位的数字量。最小值代表 GND，最大值代表基准电压减去 1LSB。基准电压源可以为 ADC 的电源电压 AVCC，外部基准电压 AVREF 或内部 1.024V/2.048V 的参考电压，通过写 ADMUX 寄存器的 REFS 位来选择。

模拟输入通道可以通过写 ADMUX 寄存器的 CHMUX 位来选择。任何 ADC 的输入引脚，外部基准电压引脚，以及内部参考电压源均可作为 ADC 的单端输入。通过置位 ADTMR 寄存器的 DIFS 可将 ADC 的输入通道切换到内部差分放大器。差分放大器相关输入源以及增益可以通过 DAPCR 寄存器设置。

通过设置 ADCSRA 寄存器的 ADEN 位即可启动 ADC，ADEN 清零时 ADC 并不耗电，因此建议在进入睡眠模式之前关闭 ADC。

ADC 转换结果为 12 位，存放与 ADC 数据寄存器 ADCH 及 ADCL 中。默认情况下转换结果为右对齐，但可通过设置 ADMUX 寄存器的 ADLAR 位变为左对齐。

如果设置为转换结果左对齐，且最高只需要 8 位的转换精度，那么只要读取 ADCH 就足够了。否则要先读取 ADCL，再读取 ADCH，以保证数据寄存器中的内容是同一次转换的结果。一旦读取 ADCL 后，数据寄存器 ADCL 和 ADCH 被锁存，读取 ADCH 后转换结果即可再更新到数据寄存器 ADCL 和 ADCH。

ADC 转换结束可以触发中断。即使转换结束发生在读取 ADCL 与 ADCH 之间，中断仍将触发。

启动一次转换

向 ADC 启动转换位 ADSC 位写“1”可以启动单次转换。在转换过程中此位保持为高，直到转换结束后被硬件清零。如果在转换过程中改变了通道，那么 ADC 会在改变通道前完成这一次转换。

ADC 转换有不同的触发源。设置 ADCSRA 寄存器的 ADC 自动触发允许位 ADATE 可以使能自动触发。设置 ADCSRB 寄存器的 ADC 触发选择位 ADTS 可以选择触发源。当所选的触发信号产生上升沿时，ADC 预分频器复位并开始转换。这提供了一个在固定时间间隔下启动转换的方法。转换结束后即使触发信号仍然存在，也不会启动一次新的转换。如果在转换过程中触发信号又产生了一个上升沿，这个上升沿也将被忽略。即使特定的中断被禁止或全局中断使能位为“0”，其中断标志仍将置位。这样可以在不产生中断的情况下触发一次转换。但是为了在下次中断事件发生时触发新的转换，必须将中断标志清零。

使用 ADC 中断标志作为触发源，可以在当前进行的转换结束后即开始下一次 ADC 转换。之后 ADC 便工作于连续转换模式，持续地进行采样并对 ADC 数据寄存器进行更新。第一次转

换是通过往 ADCSRA 寄存器的 ADSC 位写“1”来启动。在此模式下，后续的 ADC 转换不依赖于 ADC 中断标志 ADIF 是否置位。

如果使能了自动触发，置位 ADCSRA 寄存器的 ADSC 将启动单次转换。ADSC 标志还可用来检测转换是否在进行之中。不论转换是如何启动，在转换过程中 ADSC 一直为“1”。

预分频与 ADC 转换时序

在默认条件下，逐次逼近电路需要一个从 300KHz 到 3MHz 的输入时钟以获得最大精度。如果所需的转换精度低于 12 位，那么输入时钟的频率可以高于 3MHz，以达到更高的采样率。

ADC 模块包括一个预分频器，它可以由系统时钟来产生可接受的 ADC 输入时钟。预分频器通过 ADCSRA 寄存器的 ADPS 位进行设置。置位 ADCSRA 寄存器的 ADEN 将使能 ADC，预分频器开始计数。只要 ADEN 位为“1”，预分频器就持续计数，直到 ADEN 被清零。

ADCSRA 寄存器的 ADSC 被置位后，单端转换在下一个 ADC 时钟周期的上升沿开始启动。正常转换需要 15 个 ADC 时钟周期。ADC 使能（ADCSRA 寄存器的 ADEN 置位）后需要 50 个 ADC 输入时钟周期初始化模拟电路，之后才能有效进行第一次转换。

在 ADC 转换过程中，采样保持在转换启动之后的 1.5 个 ADC 输入时钟开始，而第一次 ADC 转换的结果输出则发生在启动之后的 14.5 个 ADC 输入时钟。转换结束后，ADC 结果被送入 ADC 数据寄存器，且 ADIF 标志位被置位。ADSC 同时被清零。之后软件可以再次置位 ADSC 标志或自动触发，从而启动一次新的转换。

采样通道与参考电压

ADMUX 寄存器中的 MUX 及 REFS 通过临时寄存器实现了单缓冲。CPU 可对临时寄存器进行随机访问。在转换启动之前，CPU 可随时对通道及基准源的选择进行配置。为了保证 ADC 有充足的采样时间，一旦转换开始后，就不允许通道及基准源选择的配置。在转换完成（ADCSRA 寄存器的 ADIF 置位）之后，通道及基准源的选择才会被更新。转换的开始时刻为 ADSC 置位后的下一个 ADC 输入时钟的上升沿。因此，建议用户在置位 ADSC 之后的一个 ADC 输入时钟周期内，不要操作 ADMUX 以选择新的通道及基准源。

使用自动触发时，触发事件发生的时间是不确定的。为了控制新设置对转换的影响，在更新 ADMUX 寄存器时要特别小心。若 ADATE 及 ADEN 都置位，则中断时间可以在任意时刻发生，从而自动触发，启动 ADC 的转换。如果在此期间改变 ADMUX 寄存器的内容，那么用户就无法辨别下一次转换是基于旧的配置还是新的配置。建议用户在以下安全时刻对 ADMUX 进行更新：

1）ADATE 或 ADEN 位为“0”；
2）在转换过程中，但是在触发事件发生后至少一个 ADC 输入时钟周期；
3）转换结束之后，但是在触发源的中断标志清零之前。

如果在上面所提到的任一种情况下更新 ADMUX，那么新配置将在下一次转换前生效。
选择 ADC 输入通道时须注意，在启动转换之前先选定通道，在 ADSC 置位后的一个 ADC 时钟周期之后就可以选择新的模拟输入通道，但最简单的办法是等到转换结束之后再改变通道。

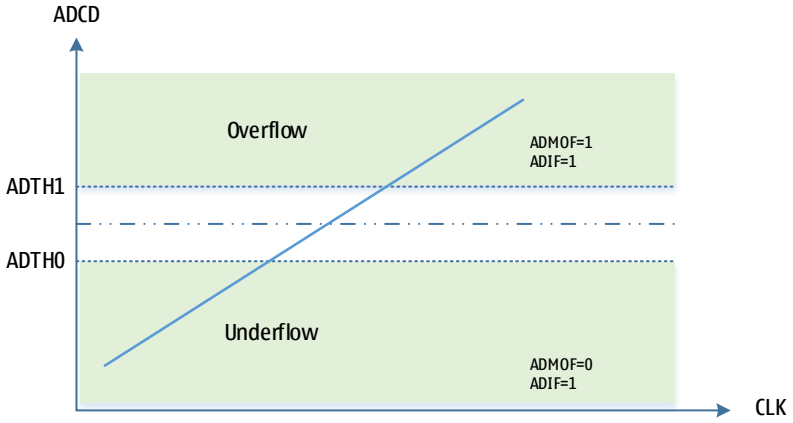
ADC 的参考电压源 V_{ref} 反映了 ADC 的转换范围。若单端通道电平超过了 V_{ref}，其转换结果将接近最大值 0xFFF。V_{ref} 可以是 AVCC，外接 AREF 引脚的电压，内部基准电压源。

使用内部基准(1.024V/2.048V/4.096V)注意事项：

芯片上电后，默认将内部基准校准为 1.024V，用户如果使用 1.024V 的内部基准，可以直接使用，无需其他操作。但如果需要使用 2.048V 或 4.096V 的内部参考电压，需要自行更新内部基准的校准值。2.048V/4.096V 的校准值在上电后被加载到寄存器 VCAL2/3(0xCE/0xCC)，在程序初始化时，将 VCAL2/3 的值读入并写入到 VCAL(0XC8)寄存器即完成校准。

自动通道监测

自动通道监测模式用于实时监测选定 ADC 输入通道的电压变化。软件通过置位 ADCSRC 寄存器的 AMEN 位使能自动通道监测功能，ADC 自动转换选定通道的电压，当转换结果在给定的溢出范围之外，将会置位 ADC 中断标志位(ADIF)，并同时停止自动监测。软件可以通过中断或查询的方式响应溢出事件。ADMSC 寄存器的 AMOF 位用于指示溢出事件的类型。ADIF 标志位在响应中断复位后自动由硬件清零；在查询模式下，可由软件写 1 清零。只有当 ADIF 清零，并通过置位 ADCSRC 寄存器的 AMEN 位，才可重新使能自动监测模式。



为克服单次 ADC 转换结果的不稳定，自动检测支持一个可配置的数字滤波功能。数字滤波通过对连续转换结果进行检测，只有在限定的连续转换次数内都得到一个一致的结果，才触发溢出事件。连续转换次数可以通过 ADMSC 寄存器的 AMFC[3:0]位设置。

自动通道监测功能通过 ADCSRC 寄存器的 AMEN 位控制。寄存器 ADT0 用于设置下溢出的阀值；ADT1 用于设置上溢出的阀值。ADT0/1 为 16 位寄存器。软件置位 AMEN 位后，将会立刻停止 ADC 当前的转换动作，并复位 ADC 控制状态，之后进入自动转换模式。

在启动自动通道检测模式前，需要设置好检测的通道以及其他相关配置。软件可以随时通过清零 AMEN 寄存器，禁止自动检测模式。

多路输入分压电路(VDS)

ADC 内部包含一个多路输入的分压模块。分压输入电压源可选来自外部 ADC 输入通道（ADC0/1/4/5）、外部参考 AVREF 或者模拟工作电源。分压模块同时输出 4/5 以及 1/5 两路电压分别到 ADC 的内部 12、13 输入通道。其中 4/5 这一路多用于 ADC 失调校准；1/5 除用于内部失调校准外，多用于电源电压检测等类似应用。分压电路相关功能主要由 ADCSRD 寄存器控制实现。

ADC 失调校准

由于制造工艺的偏差以及电路结构的固有特性， 会造成 ADC 内部比较器电路产生不同程度失调误差。因此对失调电压进行补偿，对于产生高精度的 ADC 转换结构非常关键。LGT8FX8P 芯片内部的 ADC 支持失调电压测试相关接口，可以在软件的配合下完成失调的测量和校准。

失调校准的原理：

失调校准主要是通过改变内部比较器的输入极性，在正、反两个方向测试 ADC 转换结果。由于正反两个方向失调电压也是表现为两种极性，通过这两次转换结果相减，可以得到一个中间的失调误差值。正常应用时，将转换结果根据这个失调电压进行相应的调整即可。

失调校准流程：

1. 配置 VDS 模块，将 VDS 输入源选择为模拟电源(AVCC)
2. ADC 的参考电压选择为模拟电源(AVCC)
3. ADCSRC[SPN] = 0, ADC 读取 4/5VDO 通道，转换值记录为 PVAL
4. ADCSRC[SPN] = 1, ADC 读取 4/5VDO 通道，转换值记录位 NVAL
5. 将值(NVAL – PVAL) >>1 存储到 OFR0 寄存器
6. ADCSRC[SPN] = 1, ADC 读取 1/5VDO 通道，转换结果记录为 NVAL
7. ADCSRC[SPN] = 0, ADC 读取 1/5VDO 通道，转换结果记录位 PVAL
8. 将值(NVAL – PVAL) >> 1 存储到 OFR1 寄存器
9. 设置 ADCSRC[OFEN]=1 使能失调补偿功能

特别注意：由于失调误差有正负方向， 以上数据以及运算都为有符号操作。

失调校准过程中需要改变 ADC 相关配置，因此建议失调校准在正常使用的配置之前完成。为了提高校准精度，建议 ADC 读取通道转换时采样多次滤波。

失调校准 OFR0/1 配置完成后，通过 OFEN 位使能自动失调补偿。以后的正常转换后，ADC 控制将根据 ADC 转化结果，自动使用 OFR0/1 进行补偿。

ADC 动态校准

上面介绍的失调校准方法，基于在一个测试环境和测试输入下的失调。当系统环境改变后，ADC 的失调也会随之变化。因此如果能够实现实时的校准补偿，对于克服器件随工作环境变化而导致的性能差异，提高 ADC 测量精度， 非常重要。

这里提供一种建议使用的算法，基于失调校准算法的原理，可以实现动态补偿工作环境带来的失调误差，获得一致准确的测试结果。

这种方法无需计算失调电压，也不用使能失调补偿(OFEN)。算法只需要通过 SPN 控制 ADC 转换的极性，在不同 SPN 下采样两个测量结果，两个结果中由于失调引入的误差表现为正负两种方向，因此我们可以简单的通过相加求平均的方法抵消失调产生的误差。

我们假设当在 ADC 转换时，失调引入的测试误差为 VOFS，因此控制 SPN 进行连续两次 ADC 转换，所得到的 ADC 转换结果可以表示为：

SPN = 1 时， $V_{ADC1} = V_{REL} + V_{OFS1}$

SPN = 0 时， $V_{ADC0} = V_{REL} - V_{OFS0}$

我们将两次测量结果相加，即可消除掉 V_{OFS} 对实际采样输入 V_{REL} 产生的影响。由于电路的匹配特性，V_{OFS1} 和 V_{OFS0} 可能不会完全相同，但总体上仍然可以实现补偿失调误差的效果。

动态失调补偿算法流程：

1. 根据应用需要初始化 ADC 转换参数

2. 设置 SPN=1，启动 ADC 采样，记录 ADC 采样结果为 VADC1
3. 设置 SPN=0，启动 ADC 采样，记录 ADC 采样结果为 VADC2
4. (VADC1 + VADC2) >> 1 即为本次 ADC 的转换结果

实际应用中，可以将这种算法与取样平均算法结合，可以得到更加理想的效果。

寄存器定义

ADC 寄存器列表

寄存器	地址	默认值	描述
ADCL	0x78	0x00	ADC 数据低字节寄存器
ADCH	0x79	0x00	ADC 数据高字节寄存器
ADCSRA	0x7A	0x00	ADC 控制和状态寄存器 A
ADCSRB	0x7B	0x00	ADC 控制和状态寄存器 B
ADMUX	0x7C	0x00	ADC 多路选择控制寄存器
ADCSRC	0x7D	0x01	ADC 控制和状态寄存器 C
DIDR0	0x7E	0x00	数字输入禁止控制寄存器 0
DIDR1	0x7F	0x00	数字输入禁止控制寄存器 0
DAPCR	0xDC	0x00	差分放大器控制寄存器
OFR0	0xA3	0x00	失调补偿寄存器 0
OFR1	0xA4	0x00	失调补偿寄存器 1
ADT0L	0xA5	0x00	自动监测下溢阀值低 8 位
ADT0H	0xA6	0x00	自动监测下溢阀值高 8 位
ADT1L	0xAA	0x00	自动监测上溢阀值低 8 位
ADT1H	0xAB	0x00	自动监测上溢阀值高 8 位
ADMSC	0xAC	0x01	自动监测状态和控制寄存器
ADCSRD	0xAD	0x00	ADC 控制和状态寄存器 D

ADCL – ADC 数据低字节寄存器

ADCL – ADC 数据低字节寄存器								
地址: 0x78					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name0	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0
Name1	ADC3	ADC2	ADC1	ADC0	-	-	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	描述
7:0	ADC[7:0]/ ADC[3:0]	ADC 数据低字节寄存器。 当 ADLAR 位为“0”时，ADC 输出数据在寄存器中的存放按低位对齐，即 ADCL 为 ADC[7:0]，如 Name0 所示；当 ADLAR 位为“1”时，ADC 输出数据在寄存器中的存放按高位对齐，即 ADCL 的高 4 位为 ADC[3:0]，低 4 位无意义，如 Name1 所示。

ADCH – ADC 数据高字节寄存器

ADCH – ADC 数据高字节寄存器								
地址: 0x79					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name0	-	-	-	-	ADC11	ADC10	ADC9	ADC8
Name1	ADC11	ADC10	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	描述
7:0	ADC[11:8]/ ADC[11:4]	ADC 数据低字节寄存器。 当 ADLAR 位为“0”时，ADC 输出数据在寄存器中的存放按低位对齐，即 ADCH 的低 4 位为 ADC[11:8]，高 4 位无意义，如 Name0 所示；当 ADLAR 位为“1”时，ADC 输出数据在寄存器中的存放按高位对齐，即 ADCH 为 ADC[11:4]，如 Name1 所示。

ADCSRA – ADC 控制和状态寄存器 A

ADCSRA – ADC 控制和状态寄存器 A								
地址: 0x7A					默认值: 0x05			
Bit	7	6	5	4	3	2	1	0
Name	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	1	0

Bit	Name	描述
7	ADEN	ADC 使能控制位。 当设置 ADEN 位为“1”时，ADC 被使能。 当设置 ADEN 位为“0”时，ADC 被禁止。
6	ADSC	ADC 开始转换。 在单次转换模式下，ADSC 置位将启动一次转换。在连续转换模式下，ADSC 置位将启动首次转换。
5	ADATE	ADC 自动触发使能控制位。 当设置 ADATE 位为“1”时，自动触发功能被使能。所选中触发信号的上升沿开启一次转换。触发源的选择由 ADCSRB 寄存器的 ADTS 来控制。 当设置 ADATE 位为“0”时，自动触发功能被禁止。
4	ADIF	ADC 中断标志位。 当 ADC 完成一次转换并更新数据寄存器后置位 ADIF。若 ADC 中断使能位 ADIE 为“1”且全局中断置位，ADC 中断产生。执行 ADC 中断会清零 ADIF 位，也可对该位写“1”来清零。
3	ADIE	ADC 中断使能控制位。 当设置 ADIE 位为“1”且全局中断置位时，ADC 中断被使能。 当设置 ADIE 位为“0”时，ADC 中断被禁止。

2:0	ADPS[2:0]	ADC 预分频器选择控制位。 ADPS 选择系统时钟产生 ADC 时钟的预分频因子。	
		ADPS[2:0]	预分频因子
		0	2
		1	2
		2	4
		3	8
		4	16
		5	32 (default)
		6	64
		7	128

ADCSRB – ADC 控制和状态寄存器 B

ADCSRB – ADC 控制和状态寄存器 B								
地址: 0x7B						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ACME01	ACME00	ACME11	ACME10	ACTS	ADTS2	ADTS1	ADTS0
R/W	R/W	R/W	R/W	R/W	W/0	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	描述
7	ACME01	比较器 0 负端输入选择
6	ACME00	00： 负端选择外部输入 ACIN0 01： 负端选择 ADC 多路复用输出 1X： 负端选择运放 0 的输出
5	ACME11	比较器 1 负端输入选择
4	ACME10	00： 负端选择外部输入 ACIN2 01： 负端选择 ADC 多路复用输出 1X: 负端选择运放 1 的输出
3	ACTS	AC 触发源通道选择 0 – AC0 输出作为 ADC 自动转换触发源 1 – AC1 输出作为 ADC 自动转换触发源
2:0	ADTS[2:0]	ADC 自动触发源选择控制位。 当设置 ADATE 位为“1”时，自动触发功能被使能，触发源的选择由 ADTS 来控制。当设置 ADATE 位为“0”时，ADTS 的设置无效。所选中触发信号中断标志的上升沿开启一次转换。当从一个中断标志清零的触发源切换到中断标志置位的触发源会使触发信号产生一个上升沿，如果此时 ADEN 置位，ADC 也会开启一次转换。当切换到连续转换模式（ADTS=0）时，自动触发功能被禁止。
		ADTS[2:0] 触发源 0 连续转换模式 1 比较器 0/1

		2	外部中断 0
		3	定时计数器 0 比较匹配
		4	定时计数器 0 溢出
		5	定时计数器 1 比较匹配 B
		6	定时计数器 1 溢出
		7	定时计数器 1 输入捕捉事件

ADMUX – ADC 多路选择控制寄存器

ADMUX– ADC 多路选择控制寄存器								
地址: 0x7C					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	REFS1	REFS0	ADLAR	CHMUX4	CHMUX3	CHMUX2	CHMUX1	CHMUX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial	0	0	0	0	0	0	0	0

Bit	Name	描述	
7:6	REFS[1:0]	与 ADCSRD 寄存器的 REFS2 配合用于选择 ADC 的参考电压源 通过设置 REFS 控制位来选择参考电压，若在转换过程中改变 REFS 的设置，只有等到当前的转换结束之后改变才会起作用。	
		REFS2, REFS[1:0]	参考电压选择
		0_00	AREF
		0_01	AVCC
		0_10	片内 2.048V 基准电压源
		0_11	片内 1.024V 基准电压源
		1_00	片内 4.096V 基准电压源
5	ADLAR	转换结果左对齐使能控制位。 当设置 ADLAR 位为“1”时，转换结果在 ADC 数据寄存器中为左对齐。 当设置 ADLAR 位为“0”时，转换结果在 ADC 数据寄存器中为右对齐。	
4:0	CHMUX[4:0]	ADC 输入源选择控制位。	
		CHMUX[4:0]	单端输入源
		0_0000	PC0
		0_0001	PC1
		0_0010	PC2
		0_0011	PC3
		0_0100	PC4
		0_0101	PC5
		0_0110	PE1
		0_0111	PE3
		0_1001	PC7

		0_1010	PF0	
		0_1011	PE6	
		0_1100	PE7	
		0_1110	4/5VDO	内部分压电路
		0_1000	1/5VDO	
		0_1101	IVREF	内部参考
		0_1111	AGND	模拟地
		1_XXXX	DAC0	内部 DAC 输出

ADCSRC – ADC 控制状态寄存器 C

ADCSRC– ADC 控制状态寄存器 C								
地址: 0x7D				默认值: 0x00				
Bit	7	6	5	4	3	2	1	0
Name	OFEN	-	SPN	AMEN	-	SPD	DIFS	ADTM
R/W	R/W	-	R/W	R/W	-	R/W	R/W	R/W

Bit	Name	描述
7	OFEN	1=使能失调补偿；0=关闭失调补偿
6	-	Unimplemented
5	SPN	ADC 转换输入极性控制，仅用于失调校准过程。正常时必须清零
4	AMEN	通道自动监测使能； 1：使能通道自动监测功能 0：禁止通道自动监测功能
3	-	Unimplemented
2	SPD	0=ADC 低速转换模式 1=ADC 高速转换模式，仅用于低阻抗模拟输入
1	DIFS	0 = ADC 转换来自 ADC 多路复用器 1 = ADC 转换来自内部差分放大器
0	ADTM	测试模式，从 AVREF 端口上输出内部参考电压

DIDR0 – 数字输入禁止控制寄存器 0

DIDR0– 数字输入禁止控制寄存器 0								
地址: 0x7E						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	PE3D	PE1D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
7	PE3D	1=关闭 PE3 数字输入功能
6	PE1D	1=关闭 PE1 数字输入功能
5	PC5D	1=关闭 PC5 数字输入功能
4	PC4D	1=关闭 PC4 数字输入功能

3	PC3D	1=关闭 PC3 数字输入功能
2	PC2D	1=关闭 PC2 数字输入功能
1	PC1D	1=关闭 PC1 数字输入功能
0	PC0D	1=关闭 PC0 数字输入功能

DIDR1 – 数字输入禁止控制寄存器 1

DIDR1 – 数字输入禁止控制寄存器 1								
地址: 0x7F						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	PE7D	PE6D	PE0D	C0PD	PF0D	PC7D	PD7D	PD6D
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	描述
0	PD6D	1=关闭 PD6 数字输入功能
1	PD7D	1=关闭 PD7 数字输入功能
2	PC7D	1=关闭 PC7 数字输入功能
3	PF0D	1=关闭 PF0 数字输入功能
4	C0PD	1=关闭 AC0P 数字输入功能 (LQFP48)
5	PE0D	1=关闭 PE0 数字输入功能
6	PE6D	1=关闭 PE6 数字输入功能
7	PE7D	1=关闭 PE7 数字输入功能

ADCSRD – ADC 控制寄存器 D

ADCSRD – ADC 控制寄存器 D								
地址: 0xAD						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	BGEN	REFS2	IVSEL1	IVSEL0	-	VDS2	VDS1	VDS0
R/W	R/W	R/W	R/W	R/W	-	R/W	R/W	R/W

Bit	Name	描述
7	BGEN	内部参考全局使能控制， 1=使能
6	REFS2	与 ADMUX 寄存器的 REFS 组合用于选择 ADC 转换的参考电压 请参考 ADMUX 寄存器中 REFS 的定义
5:4	IVSEL	当 ADC 的参考电压选为 VCC 或 AVREF，IVSEL 用于控制内部参考的输出电压： 00 = 1.024V 01 = 2.048V 1x = 4.096V
3	-	保留
2:0	VDS[2:0]	分压电路输入源选择 000/111 = 关闭分压电路模块 001 = ADC0 010 = ADC1 011 = ADC4

		100 = ADC5 101 = 外部参考输入(AVREF) 110 = 系统电源

DAPCR – 差分运放控制寄存器

DAPCR – 差分运放控制寄存器								
地址: 0xDC						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	DAPEN	GA1	GA0	DNS2	DNS1	DNS0	DPS1	DPS0
R/W	W/R	W/R	W/R	W/R	W/R	R/W	R/W	R/W

Bit	Name	描述
7	DAPEN	1 = 使能差分放大器; 0 = 关闭差分放大器
6:5	GA[1:0]	差分放大器增益控制 00 = x1 01 = x8 10 = x16 11 = x32
4:2	DNS[2:0]	差分放大器反向输入端输入源选择位 000 = ADC2/APN0 001 = ADC3/APN1 010 = ADC8/APN2 011 = ADC9/APN3 100 = PE0/APN4 101 = ADC 多路复用 110 = AGND 111 = 关闭差分放大器反向输入
1:0	DPS[1:0]	差分放大器正向输入端输入源选择位 00 = ADC 多路复用 01 = ADC0/APP0 10 = ADC1/APP1 11 = AGND

OFR0 – 失调补偿寄存器 0

OFR0 – 失调补偿寄存器 0								
地址: 0xA3						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	OFR0[7:0]							
R/W	W/R							

Bit	Name	描述
7:0	OFR0	失调补偿寄存器 0; OFR0 为有符号数。以二进制补码格式存储

OFR1 – 失调补偿寄存器 1

OFR1 – 失调补偿寄存器 1								
地址: 0xA4						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	OFR1[7:0]							
R/W	W/R							

Bit	Name	描述
7:0	OFR1	失调补偿寄存器 1；OFR1 为有符号数。以二进制补码格式存储

ADMSC – ADC 通道监测状态控制寄存器

ADMSC – ADC 通道监测状态控制寄存器								
地址: 0xAC						默认值: 0x01		
Bit	7	6	5	4	3	2	1	0
Name	AMOF	-	-	-	AMFC3	AMFC2	AMFC1	AMFC0
R/W	-	-	-	-	R/W	R/W	R/W	R/W

Bit	Name	描述
7	AMOF	自动监测溢出事件类型标志位；1=上溢出， 0=下溢出
6:4	-	Unimplemented
3:0	AMFC	自动监测数字滤波控制位： 0000 = 禁用配置 0001 = 一次转换，无滤波 0010 = 两次连续一致 0011 = 三次连续一致 ..... 1110 = 14 次连续一致 1111 = 15 次连续一致

ADTOL – 自动监测下溢阀值低 8 位

ADTOL – 自动监测下溢阀值低 8 位								
地址: 0xA5						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	ADTOL[7:0]							
R/W	W/R							

Bit	Name	描述
7:0	ADTOL	自动监测下溢出阀值寄存器低 8 位

ADT0H – 自动监测下溢阀值高 8 位

ADT0H – 自动监测下溢阀值高 8 位								
地址: 0xA6					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ADT0H[7:0]							
R/W	W/R							
Bit	Name	描述						
7:0	ADT0H	自动监测下溢出阀值寄存器高 8 位						

ADT1L – 自动监测上溢阀值低 8 位

ADT0L – 自动监测上溢阀值低 8 位								
地址: 0xAA					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ADT1L[7:0]							
R/W	W/R							
Bit	Name	描述						
7:0	ADT1L	自动监测上溢出阀值寄存器低 8 位						

ADT1H – 自动监测上溢阀值高 8 位

ADT1H – 自动监测上溢阀值高 8 位								
地址: 0xAB					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	ADT1H[7:0]							
R/W	W/R							
Bit	Name	描述						
7:0	ADT1H	自动监测上溢出阀值寄存器高 8 位						

VCAL – 内部参考校准寄存器

VCAL – 内部参考校准寄存器								
地址: 0xC8					默认值: 0x00			
Bit	7	6	5	4	3	2	1	0
Name	VCAL[7:0]							
R/W	W/R							
Bit	Name	描述						
7:0	VCAL	内部参考校准寄存器。上电后默认加载 1.024V 的校准值。 将其他参考电压的校准值写入此寄存器， 可实现对相关参考的校准。 比如参考配置为 2.048V 后，将 VCAL2 写入改寄存器，完成对 2.048V 内部参考的校准。						

VCAL1 – 1.024V 参考校准寄存器

VCAL1 – 1.024V 内部参考校准寄存器								
地址: 0xCD						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL1[7:0]							
R/W	R/O							
Bit	Name	描述						
7:0	VCAL1	1.024V 内部参考校准系数						

VCAL2 – 2.048V 参考校准寄存器

VCAL2 – 2.048V 内部参考校准寄存器								
地址: 0xCE						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL2[7:0]							
R/W	R/O							
Bit	Name	描述						
7:0	VCAL2	2.048V 内部参考校准系数						

VCAL3– 4.096V 参考校准寄存器

VCAL1 – 4.096V 内部参考校准寄存器								
地址: 0xCC						默认值: 0x00		
Bit	7	6	5	4	3	2	1	0
Name	VCAL3[7:0]							
R/W	R/O							
Bit	Name	描述						
7:0	VCAL3	4.096V 内部参考校准系数						

寄存器速查表

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Extended IO Register									
$F6	GUID3	GUID Byte 3							
$F5	GUID2	GUID Byte 2							
$F4	GUID1	GUID Byte 1							
$F3	GUID0	GUID Byte 0							
$F2	PMCR	PMCE	CLKFS	CLKSS	WCLKS	OSCKEN	OSCMEN	RCKEN	RCMEN
$F0	PMX2	WCE	STOSC1	STOSC0	-	-	XIEN	E6EN	C6EN
$EE	PMX0	PMXCE	C1BF4	C1AF5	C0BF3	C0AC0	SSB1	TXD6	RXD5
$ED	PMX1	-	-	-	-	-	C3AC	C2BF7	C2AF6
$EC	TCKSR	-	F2XEN	TC2XF1	TC2XF0	-	AFCKS	TC2XS1	TC2XS0
$E2	PSSR	PSS1	PSS3	-	-	-	-	PSR3	PSR1
$E1	OCPUE	PUE7	PUE6	PUE5	PUE4	PUE3	PUE2	PUE1	PUE0
$E0	HDR	-	-	HDR5	HDR4	HDR3	HDR2	HDR1	HDR0
$DE	DAPTE	DAPTE	-	-	-	-	-	-	-
$DD	DAPTR	DAPTP	DAP Trimming						
$DC	DAPCR	DAPEN	GA1	GA0	DNS2	DNS1	DNS0	DPS1	DPS0
$D8									
$D7									
$D6									
$D5									
$D4									
$D2									
$D1									
$D0									
$CF	LDOCR	WCE				PDEN	VSEL2	VSEL1	VSEL0
$CE	VCAL2	Calibration value for 2.048V internal reference							
$CD	VCAL1	Calibration value for 1.024V internal reference							
$CC	VCAL3	Calibration value for 4.096V internal reference							
$C8	VCAL	Internal Voltage Reference calibration register							
$C6	UDR	USART Data Register							
$C5	UBRRH	-	-	-	-	USART Baud Rate Register High			
$C4	UBRRL	USART Baud Rate Register Low							
$C2	UCSRC	UMSEL1	UMSEL0	UPM1	UPM0	USBS0	UCSZ01	UCSZ00	UCPOL0
$C1	UCSRB	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80
$C0	UCSRA	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0
$BD	TWAMR	TWI Address Mask							-
$BC	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE
$BB	TWDR	TWI Data							
$BA	TWAR	TWI Address							TWGCE

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$B9	TWSR	TWI Status bits					-	TWPS	
$B8	TWBR	TWI Bit Rate register							
$B6	ASSR	INTCK	-	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB
$B4	OCR2B	Timer 2 Output Compare Register B							
$B3	OCR2A	Timer 2 Output Compare Register A							
$B2	TCNT2	Timer 2 Counter Register							
$B1	TCCR2B	FOC2A	FOC2B	-	-	WGM22	CS2		
$B0	TCCR2A	COM2A1	COM2A0	COM2B1	COM2B0	-	-	WGM21	WGM20
$AF	DPS2R	-	-	-	-	DPS2E	LPRCE	TOS1	TOS0
$AE	IOCWK	IOCD7	IOCD6	IOCD5	IOCD4	IOCD3	IOCD2	IOCD1	IOCD0
$AD	ADCSRD	BGEN	REFS2	IVSEL1	IVSEL0	-	VDS2	VDS1	VDS0
$AC	ADMSC	AMOF	-	-	-	AMFC3	AMFC2	AMFC1	AMFC0
$AB	ADT1H	ADC Auto-monitor Overflow threshold high byte							
$AA	ADT1L	ADC Auto-monitor Overflow threshold low byte							
$A9	PORTE	Port Output E (for compatible with LGT8FX8D)							
$A8	DDRE	Data Direction E (for compatible with LGT8FX8D)							
$A7	PINE	Port Input E (for compatible with LGT8FX8D)							
$A6	ADT0H	ADC Auto-monitor Underflow threshold high byte							
$A5	ADT0L	ADC Auto-monitor Underflow threshold low byte							
$A4	OFR1	ADC positive offset trimming							
$A3	OFR0	ADC negative offset trimming							
$A1	DALR	DAC data register							
$A0	DACON	-	-	-	-	DACEN	DAOE	DAVS1	DAVS0
$9F	OCR3CH	Compare output register high byte of Timer3 C channel							
$9E	OCR3CL	Compare output register low byte of Timer3 C channel							
$9D	DTR3H	Dead-band register high byte of Timer3							
$9C	DTR3L	Dead-band register low byte of Timer3							
$9B	OCR3BH	Compare output register high byte of Timer3 B channel							
$9A	OCR3BL	Compare output register low byte of Timer3 B channel							
$99	OCR3AH	Compare output register high byte of Timer3 A channel							
$98	OCR3AL	Compare output register low byte of Timer3 A channel							
$97	ICR3H	Input capture register high byte of Timer3							
$96	ICR3L	Input capture register low byte of Timer3							
$95	TCNT3H	Counter register high byte of Timer3							
$94	TCNT3L	Counter register low byte of Timer3							
$93	TCCR3D	Control register D of Timer3							
$92	TCCR3C	Control register C of Timer3							
$91	TCCR3B	Control register B of Timer3							
$90	TCCR3A	Control register A of Timer3							
$8D	DTR1H	Dead-band register high byte of Timer1							
$8C	DTR1L	Dead-band register low byte of Timer1							
$8B	OCR1BH	Timer 1 Output Compare B High							

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$8A	OCR1BL	Timer 1 Output Compare B Low							
$89	OCR1AH	Timer 1 Output Compare A High							
$88	OCR1AL	Timer 1 Output Compare A Low							
$87	ICR1H	Timer 1 Input Capture High							
$86	ICR1L	Timer 1 Input Capture Low							
$85	TCNT1H	Timer 1 Counter High							
$84	TCNT1L	Timer 1 Counter Low							
$83	TCCR1D	DSX17	DSX16	DSX15	DAX14	-	-	DSX11	DSX10
$82	TCCR1C	FOC1A	FOC1B	DOC1B	DOC1A	DTEN1	-	-	-
$81	TCCR1B	ICNC1	ICES1	-	WGM13	WGM12	CS1		
$80	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	-	-	WGM11	WGM10
$7F	DIDR1	PE7D	PE6D	PE0D	C0PD	PF0D	PC7D	PD7D	PD6D
$7E	DIDR0	PE3D	PE1D	PC5D	PC4D	PC3D	PC2D	PC1D	PC0D
$7D	ADCSRC	OFEN	-	SPN	AMEN	-	SPD	DIFS	ADTM
$7C	ADMUX	REFS1	REFS0	ADLAR	CHMUX				
$7B	ADCSRB	CME01	CME00	CME11	CME10	-	ADTS		
$7A	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS		
$79	ADCH	ADC Data High							
$78	ADCL	ADC Data Low							
$76	DIDR2	-	PB5D	-	-	-	-	-	-
$75	IVBASE	Interrupt Vector Base Address							
$74	PCMSK4								
$73	PCMSK3	PCINT[39:32]							
$71	TIMSK3			ICIE3	-	OCIE3C	OCIE3B	OCIE3A	TOIE3
$70	TIMSK2	-	-	-	-	-	OCIE2B	OCIE2A	TOIE2
$6F	TIMSK1	-	-	ICIE1	-	-	OCIE1B	OCIE1A	TOIE1
$6E	TIMSK0	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0
$6D	PCMSK2	PCINT[23:16]							
$6C	PCMSK1	PCINT[15:8]							
$6B	PCMSK0	PCINT[7:0]							
$69	EICRA	-	-	-	-	ISC11	ISC10	ISC01	ISC00
$68	PCICR	-	-	-	PCIE4	PCIE3	PCIE2	PCIE1	PCIE0
$67	RCKCAL	RC32K Calibration							
$66	RCMCAL	RC32M Calibration							
$65	PRR1	-	-	PRWDT	-	PRTIM3	PREFL	PRPCI	-
$64	PRR0	PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUART0	PRADC
$62	VDTCR	WCE	SWR	-	VDTS			VDREN	VDTEN
$61	CLKPR	WCE	CKOE1	CKOE0	-	CLKPS			
$60	WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0
DirectIO Register									
$5F	SREG	I	T	H	S	V	N	Z	C
$5E	SPH	Stack Point High							

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$5D	SPL	Stack Point Low							
$5C	E2PD3	E2PCTL Data register byte 3							
$5B	C1TR	AC1 trimming data							
$5A	E2PD1	E2PCTL Data register byte1							
$59	DSAH	DSA[31:16] access port of uDSC							
$58	DSAL	DSA[15:0] access port of uDSC							
$57	E2PD2	E2PCTL Data register byte 2							
$56	ECCR	WEN	EEN	ERN	SWM	CP1	CP0	ECS1	ECS0
$55	MCUCR	FWKEN	FPDEN	SWR	PUD	IRLD	IFAIL	IVSEL	WCE
$54	MCUSR	SWDD	-	-	OCDRF	WDRF	BORF	EXTRF	PORF
$53	SMCR	-	-	-	-	SM			SE
$52	C0TR	AC0 Trimming register							
$51	C0XR	-	C0OE	C0HYSE	C0PS0	C0WKE	C0FEN	C0FS1	C0FS0
$50	C0SR	C0D	C0BG	C0O	C0I	C0IE	C0IC	C0IS	
$4F	DTR0	TC0 Dead-band timing control register							
$4E	SPDR	SPI Data register							
$4D	SPSR	SPIF	WCOL	-	-	-	DUAL	-	SPI2X
$4C	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR	
$4B	GPIOR2	General Purpose Register 2							
$4A	GPIOR1	General Purpose Register 1							
$49	TCCR0C	DSX07	DSX06	DSX05	DSX04	-	-	DSX01	DSX00
$48	OCR0B	Timer 0 Output Compare Register B							
$47	OCR0A	Timer 0 Output Compare Register A							
$46	TCNT0	Timer 0 Counter							
$45	TCCR0B	FOC0A	FOC0B	OC0AS	DTEN0	WGM02	CS02	CS01	CS00
$44	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	DOC0B	DOC0A	WGM01	WGM00
$43	GTCCR	TSM	-	-	-	-	-	PSRASY	PSRSYNC
$42	EEARH	E2PCTL Address High							
$41	EEARL	E2PCTL Address Low							
$40	E2PD0	E2PCTL Data byte 0							
$3F	EECR	EEPM2	EEPM2	EEPM1	EEPM0	EERIE	EEMWE	EEWE	EERE
$3E	GPIOR0	General Purpose Register 0							
$3D	EIMSK	-	-	-	-	-	-	INT1	INT0
$3C	EIFR	-	-	-	-	-	-	INTF1	INTF0
$3B	PCIFR	-	-	-	-	PCIF3	PCIF2	PCIF1	PCIF0
$3A	C1XR	-	C1OE	C1HYSE	C1PS0	C1WKE	C1FEN	C1FS1	C1FS0
$39	SPFR	RDFULL	RDEMPT	RDPTR1	RDPTR0	WRFULL	WREMPT	WRPTR1	WRPTR0
$38	TIFR3	-	-	ICF3	-	-	OCF3B	OCF3A	TOV3
$37	TIFR2	-	-	-	-	-	OCF2B	OCF2A	TOV2
$36	TIFR1	-	-	ICF1	-	-	OCF1B	OCF1A	TOV1
$35	TIFR0	-	-	-	-	-	OCF0B	OCF0A	TOV0
$34	PORTF	Port Output of Group F							

Addr	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$33	DDRF	Data Direction of Group F							
$32	PINF	Port Input of Group F							
$31	DSDY	DSDY access port of uDSC							
$30	DSDX	DSDX access port of uDSC							
$2F	C1SR	C1D	C1BG	C1O	C1I	C1IE	C1IC	C1IS	
$2E	PORTE	Port Output of Group E							
$2D	DDRE	Data Direction of Group E							
$2C	PINE	Port Input of Group E							
$2B	PORTD	Port Output of Group D							
$2A	DDRD	Data Direction of Group D							
$29	PIND	Port Input of Group D							
$28	PORTC	Port Output of Group C							
$27	DDRC	Data Direction of Group C							
$26	PINC	Port Input of Group C							
$25	PORTB	Port Output of Group B							
$24	DDRB	Data Direction of Group B							
$23	PINB	Port Input of Group B							
$22	DSSD	DSSD access port of uDSC							
$21	DSIR	Instruction regiter of uDSC							
$20	DSCR	DSUEN	MM	D1	D0	-	DSN	DSZ	DSC

指令集速查表

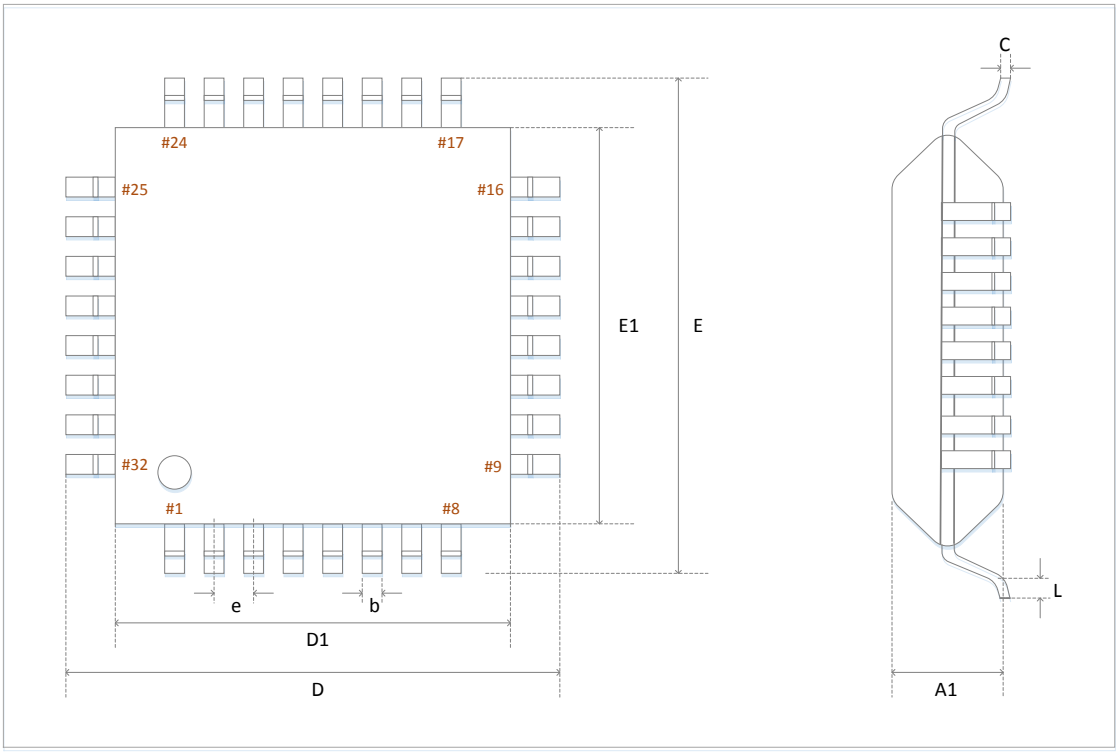
指令	操作数	描述	操作	标记位	周期
算术逻辑运算指令					
ADD	R_d, R_r	寄存器相加	$R_d \leftarrow R_d + R_r$	Z,C,N,V,H	1
ADC	R_d, R_r	带进位的寄存器相加	$R_d \leftarrow R_d + R_r + C$	Z,C,N,V,H	1
ADIW	R_{dl}, K	立即数与字相加	$R_{dh}:R_{dl} \leftarrow R_{dh}:R_{dl} + K$	Z,C,N,V,S	1
SUB	R_d, R_r	寄存器相加减	$R_d \leftarrow R_d - R_r$	Z,C,N,V,H	1
SUBI	R_d, K	寄存器减常数	$R_d \leftarrow R_d - K$	Z,C,N,V,H	1
SBC	R_d, R_r	带借位的寄存器相加减	$R_d \leftarrow R_d - R_r - C$	Z,C,N,V,H	1
SBCI	R_d, K	带借位的寄存器减常数	$R_d \leftarrow R_d – K - C$	Z,C,N,V,H	1
SBIW	R_{dl}, K	立即数与字相减	$R_{dh}:R_{dl} \leftarrow R_{dh}:R_{dl} - K$	Z,C,N,V,S	1
AND	R_d, R_r	逻辑与	$R_d \leftarrow R_d \& R_r$	Z,N,V	1
ANDI	R_d, K	寄存器逻辑与常数	$R_d \leftarrow R_d \& K$	Z,N,V	1
OR	R_d, R_r	逻辑或	$R_d \leftarrow R_d \mid R_r$	Z,N,V	1
ORI	R_d, K	寄存器逻辑或常数	$R_d \leftarrow R_d \mid K$	Z,N,V	1
EOR	R_d, R_r	寄存器异或	$R_d \leftarrow R_d \oplus R_r$	Z,N,V	1
COM	R_d	反码	$R_d \leftarrow \$FF - R_d$	Z,C,N,V	1
NEG	R_d	2 禁制补码	$R_d \leftarrow \$00 - R_d$	Z,C,N,V,H	1
SBR	R_d, K	设置寄存器中的位	$R_d \leftarrow R_d \vee K$	Z,N,V	1
CBR	R_d, K	清寄存器中的位	$R_d \leftarrow R_d \vee (\$FF – K)$	Z,N,V	1
INC	R_d	递增	$R_d \leftarrow R_d + 1$	Z,N,V	1
DEC	R_d	递减	$R_d \leftarrow R_d - 1$	Z,N,V	1
TST	R_d	测试为 0 或负数	$R_d \leftarrow R_d \& R_d$	Z,N,V	1
CLR	R_d	清寄存器	$R_d \leftarrow R_d \oplus R_d$	Z,N,V	1
SER	R_d	寄存器全设置为 1	$R_d \leftarrow \$FF$	None	1
MUL	R_d, R_r	无符号乘法	R_1: $R_0 \leftarrow R_d \times R_r$	Z,C	1
MULS	R_d, R_r	有符号乘法	R_1: $R_0 \leftarrow R_d \times R_r$	Z,C	1
MULSU	R_d, R_r	有符号数乘无符号数	R_1: $R_0 \leftarrow R_d \times R_r$	Z,C	1
FMUL	R_d, R_r	无符号乘法，移位	R_1: $R_0 \leftarrow (R_d \times R_r) << 1$	Z,C	1
FMULS	R_d, R_r	有符号乘法，移位	R_1: $R_0 \leftarrow (R_d \times R_r) << 1$	Z,C	1
FMULSU	R_d, R_r	有符号数乘无符号数，移位	R_1: $R_0 \leftarrow (R_d \times R_r) << 1$	Z,C	1
跳转指令					
RJMP	K	相对跳转	PC ← PC + K + 1	None	1
IJMP		间接跳转（到 Z 指向地址）	PC ← Z	None	2
JMP	K	直接跳转	PC ← K	None	2
RCALL	K	相对地址子程序调用	PC ← PC + K + 1	None	1
ICALL		间接子程序调用（Z 指向地址）	PC ← Z	None	2
CALL	K	直接子程序调用	PC ← K	None	2
RET		子程序返回	PC ← Stack	None	2
RETI		中断返回	PC ← Stack	I	2

指令	操作数	描述	操作	标记位	周期
跳转指令（续）					
CPSE	R_d, R_r	相等即跳转	If(R_d=R_r) PC ← PC + 2 or 3	None	1/2
CP	R_d, R_r	比较	R_d - R_r	Z,N,V,C,H	1
CPC	R_d, R_r	带进位比较	R_d - R_r – C	Z,N,V,C,H	1
CPI	R_d, K	与立即数比较	R_d - K	Z,N,V,C,H	1
SBRC	R_r, b	位为 0 即跳过下一条指令	If(R_r(b)=0) PC ← PC + 2 or 3	None	1/2
SBRS	R_r, b	位为 1 即跳过下一条指令	If(R_r(b)=1) PC ← PC + 2 or 3	None	1/2
SBIC	P, b	I/O 位为 0 即跳过下一条指令	If(P(b)=0) PC ← PC + 2 or 3	None	1/2
SBIS	P, b	I/O 位为 1 即跳过下一条指令	If(P(b)=1) PC ← PC + 2 or 3	None	1/2
BRBS	s, k	状态标记为 1 即跳转	If(SREG(S)=1) PC ← PC + K + 1	None	1/2
BRBC	s, k	状态标记为 0 即跳转	If(SREG(S)=0) PC ← PC + K + 1	None	1/2
BREQ	k	相等即跳转	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	不等即跳转	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	进位则跳转	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	无进位则跳转	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	不小于则跳转	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	小于则跳转	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	为负则跳转	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	为正则跳转	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	有符号的不小于即跳转	if (N⊕V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	有符号的小于 0 即跳转	if (N⊕V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	半进位为 1 则跳转	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	半进位为 0 则跳转	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	T 置位则跳转	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	T 清零则跳转	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	溢出则跳转	f (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	不溢出则跳转	f (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	全局中断使能则跳转	f (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	全局中断禁止则跳转	f (I = 0) then PC ← PC + k + 1	None	1/2
数据传输指令					
MOV	Rd, Rr	寄存器之间移动数据	Rd ← Rr	None	1
MOVW	Rd, Rr	移动一个字的数据	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	加载立即数	Rd ← K	None	1
LD	Rd, X	间接加载	Rd ← (X)	None	1/2
LD	Rd, X+	间接加载，地址递增	Rd ← (X), X ← X + 1	None	1/2
LD	Rd, -X	地址递减，间接加载	X ← X - 1, Rd ← (X)	None	1/2
LD	Rd, Y	间接加载	Rd ← (Y)	None	1/2
LD	Rd, Y+	间接加载，地址递增	Rd ← (Y), Y ← Y + 1	None	1/2
LD	Rd, -Y	地址递减，间接加载	Y ← Y - 1, Rd ← (Y)	None	1/2
LDD	Rd, Y+q	带偏移量的间接加载	Rd ← (Y + q)	None	1/2
LD	Rd, Z	间接加载	Rd ← (Z)	None	1/2

LD	Rd, Z+	间接加载，地址递增	Rd ← (Z), Z ← Z+1	None	1/2
LD	Rd, -Z	地址递减，间接加载	Z ← Z - 1, Rd ← (Z)	None	1/2
LDD	Rd, Z+q	带偏移量的间接加载	Rd ← (Z + q)	None	1/2
LDS	Rd, k	直接从 SRAM 中加载	Rd ← (k)	None	2
ST	X, Rr	间接存储	(X) ← Rr	None	1
ST	X+, Rr	间接存储，地址递增	(X) ← Rr, X ← X + 1	None	1
ST	-X, Rr	地址递减，间接存储	X ← X - 1, (X) ← Rr	None	1
ST	Y, Rr	间接存储	(Y) ← Rr	None	1
ST	Y+, Rr	间接存储，地址递增	(Y) ← Rr, Y ← Y + 1	None	1
ST	-Y, Rr	地址递减，间接存储	Y ← Y - 1, (Y) ← Rr	None	1
STD	Y+q, Rr	带偏移量的间接存储	(Y + q) ← Rr	None	1
ST	Z, Rr	间接存储	(Z) ← Rr	None	1
ST	Z+, Rr	间接存储，地址递增	(Z) ← Rr, Z ← Z + 1	None	1
ST	-Z, Rr	地址递减，间接存储	Z ← Z - 1, (Z) ← Rr	None	1
STD	Z+q, Rr	带偏移量的间接存储	(Z + q) ← Rr	None	1
STS	k, Rr	直接存储到 SRAM 中	(k) ← Rr	None	2
LPM		加载程序空间数据	R0 ← (Z)	None	2
LPM	Rd, Z	加载程序空间数据	Rd ← (Z)	None	2
LPM	Rd, Z+	加载程序数据，地址递增	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, Z+	间接加载，地址递增	Rd ← (Z), Z ← Z+1	None	1
LD	Rd, -Z	地址递减，间接加载	Z ← Z - 1, Rd ← (Z)	None	1
LDD	Rd, Z+q	带偏移量的间接加载	Rd ← (Z + q)	None	1
LDS	Rd, k	直接从 SRAM 中加载	Rd ← (k)	None	2
IN	Rd, P	读端口	Rd ← P	None	1
OUT	P, Rr	写端口	P ← Rr	None	1
PUSH	Rr	压栈	STACK ← Rr	None	1
POP	Rd	出栈	Rd ← STACK	None	1/2
SBI	P, b	设置 IO 寄存器	I/O(P, b) ← 1	None	1
CBI	P, b	清零 IO 寄存器	I/O(P, b) ← 0	None	1
LSL	Rd	逻辑左移	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	逻辑右移	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z	1
ROL	Rd	包含进位的循环左移	Rd(0)←C, Rd(n+1) ← Rd(n), C←Rd(7)	Z	1
ROR	Rd	包含进位的循环右移	Rd(7)←C, Rd(n) ← Rd(n+1), C←Rd(0)	Z	1
ASR	Rd	算术右移	Rd(n) ← Rd(n+1), n=0:6	Z	1
SWAP	Rd	位交换	Rd(3:0) ← Rd(7:4), Rd(7:4) ← Rd(3:0)	None	1
BSET	s	设置状态位	SREG(s) ← 1	SREG(s)	1
BCLR	s	清零状态位	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	存储到 T 位	T ← Rr(b)	T	1
BLD	Rd, b	读出 T 位到寄存器	Rd(b) ← T	None	1
SEC		设置进位标志	C ← 1	C	1
CLC		清楚进位标志	C ← 0	C	1

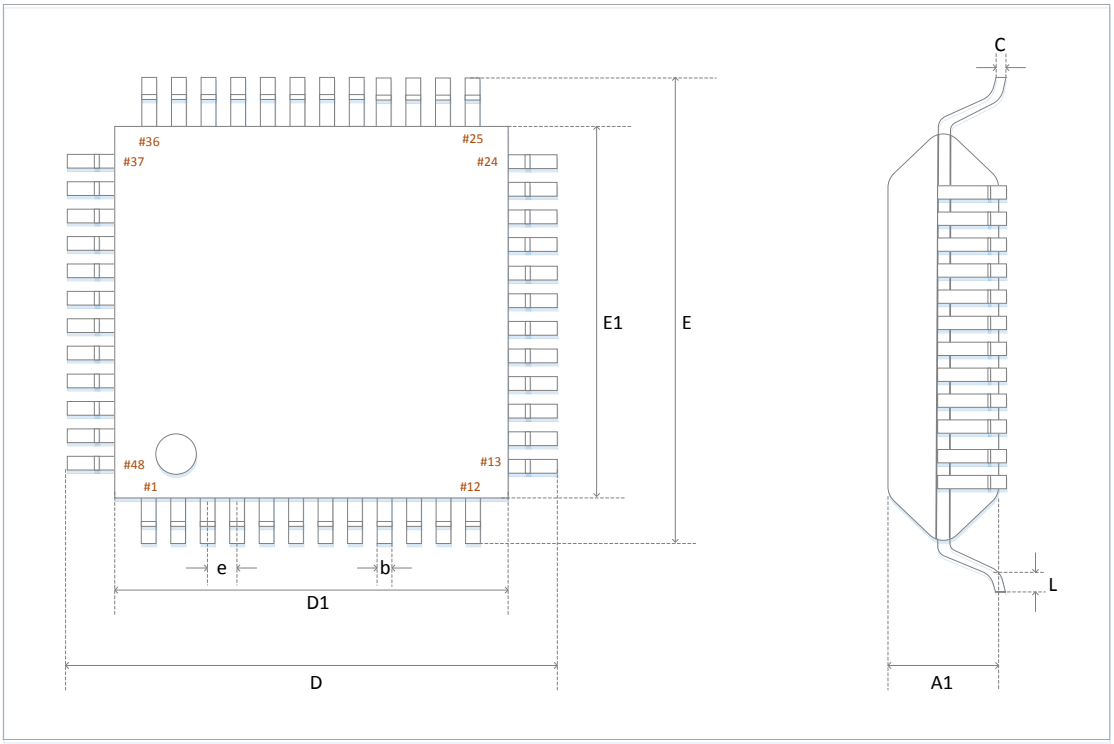
SEN		设置负数标志	N ← 1	N	1
CLN		清除负数标志	N ← 0	N	1
SEZ		设置零标志	Z ← 1	Z	1
CLZ		清除零标志	Z ← 0	Z	1
SEI		使能全局中断	I ← 1	I	1
CLI		禁制全局中断	I ← 0	I	1
SES		设置符号测试标志	S ← 1	S	1
CLS		清除符号测试标志	S ← 0	S	1
SEV		设置二进制补码溢出标志	V ← 1	V	1
CLV		清除二进制补码溢出标志	V ← 0	V	1
SET		设置 T 位（SREG）	T ← 1	T	1
CLT		清除 T 位（SREG）	T ← 0	T	1
MCU 控制指令					
NOP		空指令		None	1
SLEEP		进入休眠模式		None	1
WDR		看门狗复位		None	1
BREAK		软断点	仅用于调试目的	None	N/A
NOP		空指令		None	1
SLEEP		进入休眠模式		None	1

封装参数



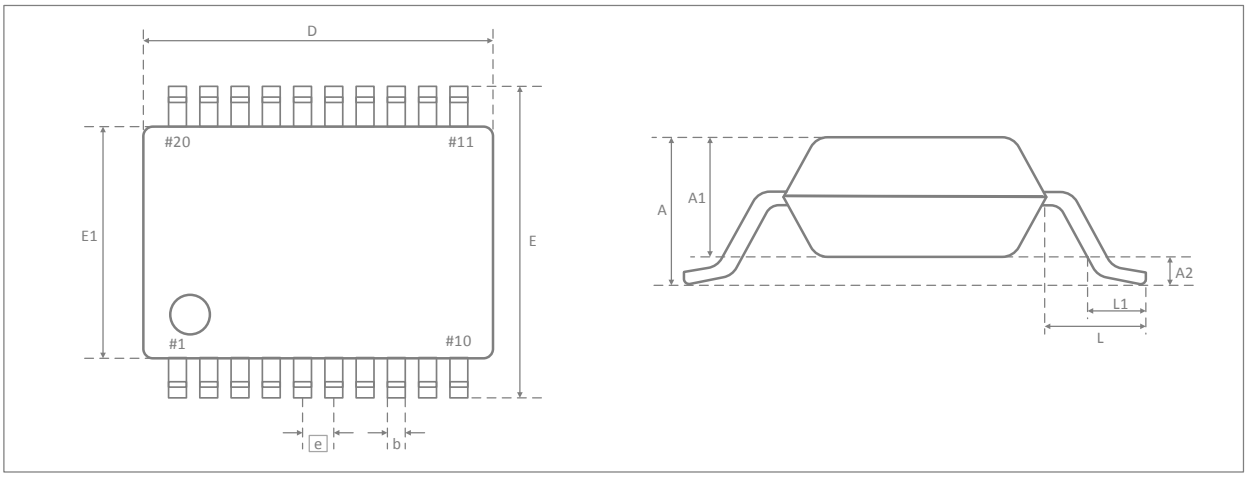
LQFP32 通用尺寸定义

字符代号	最小值	典型值	最大值	单位
D	8.90	9.00	9.10	mm
D1	6.90	7.00	7.10	mm
b	0.2	0.30	0.4	mm
e	0.75	0.80	0.85	mm
E	8.90	9.00	9.10	mm
E1	6.90	7.00	7.10	mm
C	-	0.10	-	mm
L	0.55	0.60	0.65	mm
A1	-	1.40	-	mm



LQFP48 通用尺寸定义

字符代号	最小值	典型值	最大值	单位
D	8.80	9.00	9.20	mm
D1	6.80	7.00	7.20	mm
b	0.17	0.22	0.27	mm
e	-	0.50BSC	-	mm
E	8.80	9.00	9.20	mm
E1	6.80	7.00	7.20	mm
C	0.09	-	0.2	mm
L	0.45	0.60	0.75	mm
A1	1.35	1.40	1.45	mm



SSOP20L 通用尺寸定义

字符代号	最小值	典型值	最大值	单位
D	6.90	7.20	7.50	mm
A2	0.03	0.05	0.07	mm
b	0.22	0.30	0.38	mm
e	-	0.65	-	mm
E	7.40	7.80	8.20	mm
E1	5.00	5.30	5.60	mm
L1	0.55	-	0.95	mm
L	-	-	-	mm
A1	-	2.0	-	mm

版本历史

V1.0.3 2017/6/23	增加 SSOP20 封装定义 更新 TMR3 中断标记位的操作说明
V1.0.2 2017/5/15	更新 TMR0/TRM1/TMR3 中关于自动 PWM 关闭和重启的说明 更新 SPI 章节中对 SPI 中断处理的说明以及更新 SPFR 寄存器的说明
V1.0.1 2017/2/13	删除 I2C1 部分，此功能不可用 完善了部分寄存器的定义
V1.0.0 2016/12/29	初始版本