TB3212

TCD 入门

简介

AVR®单片机具有强大的定时器，可适用于广泛的应用，从信号测量到事件同步和波形发生。

定时器/计数器类型 D（TCD）是 tinyAVR® 1 系列器件中的 12 位定时器。 TCD 旨在满足在常见嵌入式应用（例如电机控制或开关电源）中具有多种波形生成功能的快速定时器的需求。 TCD 的一个关键特性是，除了从主时钟运行外，它还可以设置为直接从内部 20 MHz 振荡器（OSC20M）运行或与外部时钟源一起使用。 此外，它具有多种可配置的波形发生模式。

本文档描述了一些 TCD 工作模式，强调了此计时器的特殊性并提供了初始化代码片段。欲更深入地理解有关功能，请参阅 tinyAVR® 1 系列数据手册。该文档包括两个特定用例：

- **生成互补驱动信号：**
  初始化定时器以生成两个互补信号，频率为 50 kHz，死区时间为 100 ns.
- **使用输入事件控制同步信号：**
  初始化定时器以产生 4 个 PWM 信号，频率为 10 kHz，占空比约为 50%，成对同步。 配置输入通道以进行故障检测。

注：代码示例使用 ATtiny817 Xplained Mini（ATTINY817-XMINI）开发。

目录

1. 相关器件

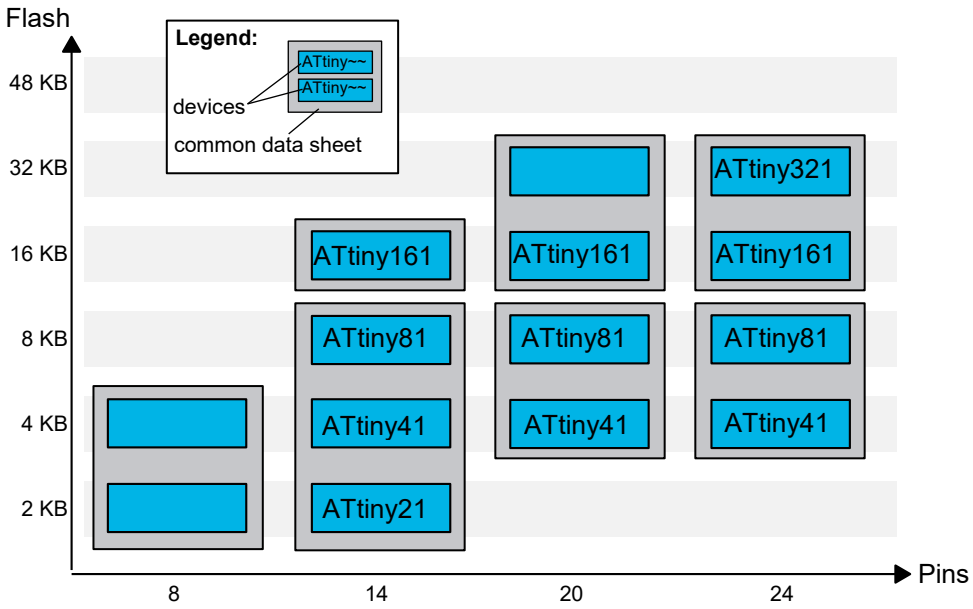
本章列出了本文档的相关器件。

1.1 tinyAVR® 1 系列

下图显示了 tinyAVR®1 系列器件，列出了引脚数类型和存储器大小：

- 无需修改代码即可实现向上垂直迁移，因为上方器件引脚兼容并提供相同或更多功能。由于下方器件可能缺少某些外设，因此向下迁移可能需要修改代码。
- 向左水平迁移，引脚数减少，因此可用功能也相应减少。

图 1-1：tinyAVR® 1 系列概览

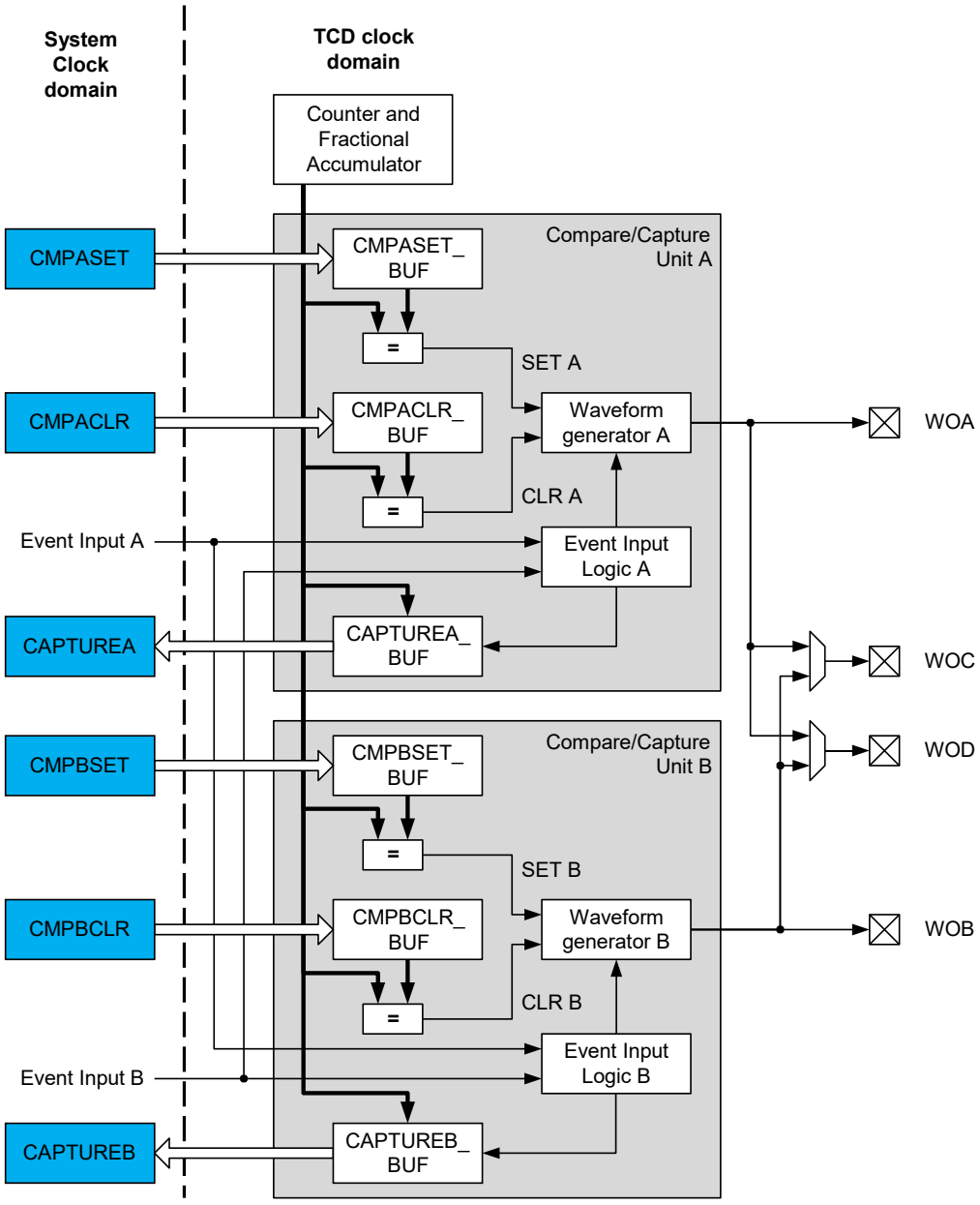


具有不同闪存大小的器件通常也具有不同的 SRAM 和 EEPROM。

2. 概述

TCD 是一种高性能波形控制器，由异步计数器，预分频器，比较逻辑，捕捉逻辑和控制逻辑组成。 TCD 包含一个可以在与系统时钟异步的时钟上运行的计数器。 它包含比较逻辑，可以生成具有可选死区时间的两个独立输出。 它连接到事件系统以进行捕捉和确定性故障控制。 定时器/计数器可以在比较匹配和溢出时产生中断和事件。

图 2-1：TCD 框图



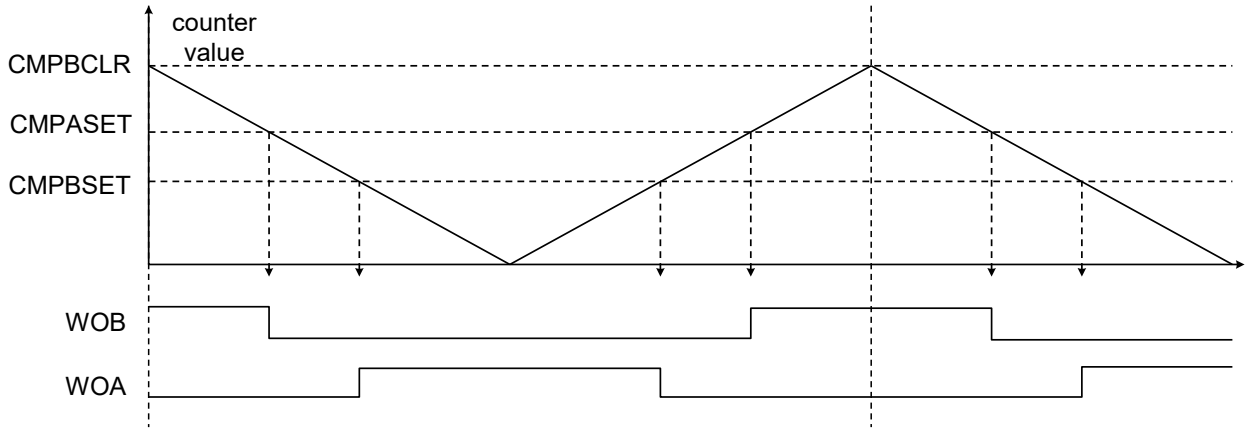
TCD 内核与系统时钟异步。定时器/计数器由两个比较/捕捉单元组成，每个单元具有单独的波形输出。此外，还有两个额外的波形输出，可以等于其中一个单元的输出。比较寄存器 CMPxSET 和 CMPxCLR 存储在各自的寄存器（TCD.CMPxSET 和 TCD.CMPxCLR）中，该寄存器由低字节和高字节组成。 写入寄存器后，寄存器与 TCD 域同步。 在正常操作期间，计数器值不断与比较寄存器进行比较。 这用于生成中断和事件。

TCD 可以使用十种不同输入模式的输入事件，可分别为两个输入事件选择不同的模式（见图 4-6）。输入模式定义输入事件将如何影响输出

3. 产生互补驱动信号

8 位单片机通常用于开关电源。该用例说明了如何配置 TCD 以生成功率 MOSFET 晶体管的互补波形。 在该示例中，TCD 实例被配置为生成具有 50 kHz 频率和 100 ns 死区时间的两个互补信号。

图 3-1：双斜坡模式



- 最大计数器值存储在CMPBLCR寄存器中（见图3-1）。
- 当TCD计数器倒计数并与CMPASET值匹配时， WOA输出有效。
- 当TCD计数器向上计数并与CMPASET值匹配时，WOA被清除。
- 当TCD计数器向上计数并与CMPBSET值匹配时，WOB输出有效。
- 当TCD计数器倒计数并与CMPBSET值匹配时，WOB被清除。

1. 使用 CTRLB 寄存器将波形生成模式设置为双斜坡。

```
TCD0.CTRLB = TCD_WGMODE_DS_gc;
```

图 3-2：CTRLB 寄存器

Bit	7	6	5	4	3	2	1	0
							WGMODE[1:0]	
Access							R/W	R/W
Reset							0	0

bits 1:0 WGMODE[1:0]: Waveform Generation Mode bits
These bits select the waveform generation.

Value	Name	Description
0x0	ONERAMP	One Ramp mode
0x1	TWORAMP	Two Ramp mode
0x2	FOURRAMP	Four Ramp mode
0x3	DS	Dual-Slope mode

2. 信号的周期必须从下面的公式中推导出来并写入 CMPBCLR 寄存器。

$$f_{signal} = \frac{f_{CLK}}{Prescaler_{CNT} \times 2 \times (CMPBCLR + 1)}$$

外设时钟将设置为 20 MHz 内部振荡器，计数器预分频器将设置为 1。

$$CMPBCLR = \frac{f_{CLK}}{Prescaler_{CNT} \times 2 \times f_{signal}} - 1 = \frac{20 \times 10^6}{1 \times 2 \times 50 \times 10^3} - 1 \cong 200 = 0xC8$$

```
TCD0.CMPBCLR = 0xC8;
```

3. 要设定死区时间，应确定计数周期，以及与死区时间匹配的计数周期数。

$$CNT_{period} = \frac{1}{f_{CNT}} = \frac{Prescaler_{CNT}}{f_{CLK}} = \frac{1}{20 \times 10^6} = 50ns$$

$$Dead\ time = 2 \times CNT_{period} = 100ns$$

假设用户希望两个信号具有相同的占空比。

$$CMPBSET = \frac{CMPBCLR}{2} + \frac{Dead\ time}{2} = 0x65$$

$$CMPASET = \frac{CMPBCLR}{2} - \frac{Dead\ time}{2} = 0x63$$

```
TCD0.CMPBSET = 0x65;
TCD0.CMPASET = 0x63;
```

4. 在使能定时器之前，必须验证 STATUS 寄存器的 ENRDY 位的值为“1”。

   当可安全启动定时器时，硬件会将该位置 1。此机制可防止 TCD 时域与系统时域之间的同步问题。

```
while(!(TCD0.STATUS & TCD_ENRDY_bm))
{
    ;
}
```

图 3-3：STATUS 寄存器

Bit	7	6	5	4	3	2	1	0
	PWMACTB	PWMACTA					CMDRDY	ENRDY
Access	R/W	R/W					R	R
Reset	0	0					0	0

5. ENRDY位置1后，可以通过CTRLA寄存器使能定时器。

   从同一寄存器，也可以设置时钟源和预分频器。CTRLA中除ENABLE位外的所有位都是使能保护的，它们只能在写入操作之前将ENABLE设置为“0”时写入。

```
TCD0.CTRLA = TCD_CLKSEL_20MHZ_gc
           | TCD_CNTPRES_DIV1_gc
           | TCD_ENABLE_bm;
```

图 3-4：CTRLA 寄存器

Bit	7	6	5	4	3	2	1	0
		CLKSEL[1:0]		CNTPRES[1:0]		SYNCPRES[1:0]		ENABLE
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset		0	0	0	0	0	0	0

bits 6:5 CLKSEL[1:0]: Clock Select bits
The clock select bits select the clock source of the TCD clock.

Value	Description
0x0	OSC20M
0x1	Reserved
0x2	External clock
0x3	System clock

bits 4:3 CNTPRES[1:0]: Counter Prescaler bits
The Counter Prescaler bits select the division factor of the TCD counter clock.

Value	Description
0x0	Division factor 1
0x1	Division factor 4
0x2	Division factor 32
0x3	Reserved

6. 要使能输出通道，应将 FAULTCTRL 寄存器中的某些位置 1。

   该寄存器受配置变更保护（CCP）。 因此，在写入 FAULTCTRL 寄存器之前，必须将一个密钥写入 CPU 的 CCP 寄存器。 在本例中，仅启用通道 A 和 B。

```
void TCD0_enableOutputChannels(void)
{
        CPU_CCP = CCP_IOREG_gc;

        TCD0.FAULTCTRL = TCD_CMPAEN_bm
                       | TCD_CMPBEN_bm;
}
```

图 3-5：FAULTCTRL 寄存器

Bit	7	6	5	4	3	2	1	0
	CMPDEN	CMPCEN	CMPBEN	CMPAEN	CMPD	CMPC	CMPB	CMPA
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	x	x	x	x	x	x	x	x

注：复位后，FAULTCTRL 寄存器从熔丝加载其值。

7. 在目标单片机中，TCD 输出通道 A 和 B 链接到 PA4 和 PA5。这些引脚必须配置为输出：

```
PORTA.DIR |= PIN4_bm
           | PIN5_bm;
```

该应用将配置 TCD 实例以生成两个互补信号，频率为 50 kHz，死区时间为 100 ns。

View Code Example on GitHub
Click to browse repository

提示：完整代码示例也可以在附录部分中找到。

4. 使用输入事件控制同步信号

在电机控制应用中，通常需要使用完全同步的信号。此外，存在一小部分应用，出于驱动目的，需要比引脚可提供的电流略大一点点的电流。 在这些情况下，如果可以在多个引脚上映射相同信号就能够减少物料清单。

在大多数嵌入式系统中，有不同的反馈回路和故障控制组件，可确保一切正常。出于可靠性原因，最好将这些反馈机制设计为不使用 MCU 内核。TCD 有两个事件输入，可用于监视其他外设和组件的活动，并相应地更改输出。

在此示例中，TCD 实例将配置为生成 4 个 PWM 信号，其频率为 10 kHz，占空比约为 50%，成对同步。如果输入通道上出现故障信号，定时器将停止并等待，直到信号变为“安全”状态（未检测到故障）。

1. 定时器将由 CTRLB 寄存器配置（与前一个实例一样），但这次是在四斜坡模式下：

```
TCD0.CTRLB = TCD_WGMODE_FOURRAMP_gc;
```

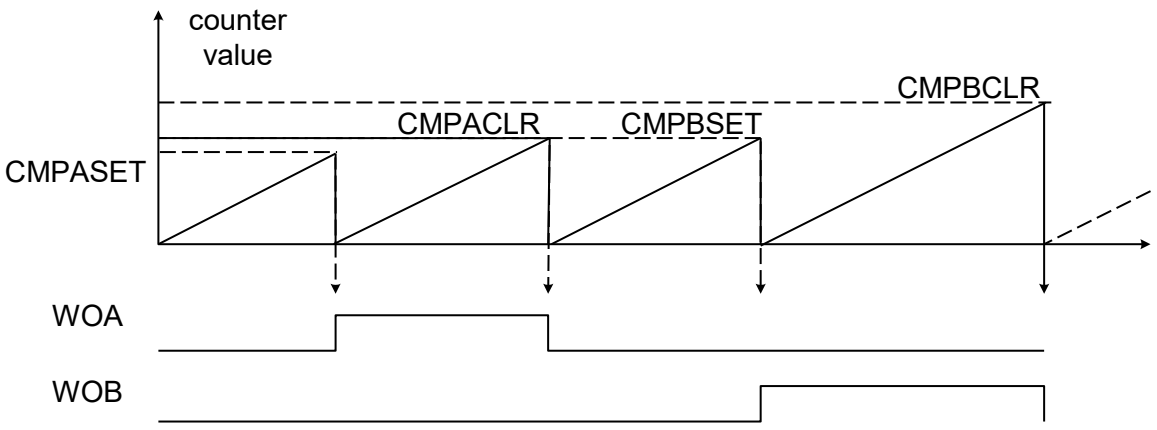
图 4-1：CTRLB 寄存器

Bit	7	6	5	4	3	2	1	0
							WGMODE[1:0]	
Access							R/W	R/W
Reset							0	0

bits 1:0 WGMODE[1:0]: Waveform Generation Mode bits
These bits select the waveform generation.

Value	Name	Description
0x0	ONERAMP	One Ramp mode
0x1	TWORAMP	Two Ramp mode
0x2	FOURRAMP	Four Ramp mode
0x3	DS	Dual-Slope mode

图 4-2：四斜坡模式



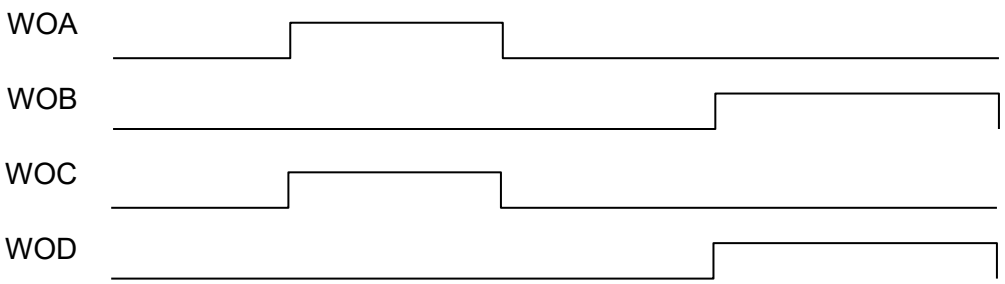
2. 默认情况下，通道 C 和 D 链接到通道 A。为了使信号成对同步，CTRLC 寄存器中的 CMPDSEL 位必须设置为将通道 D 链接到通道 B（见图 4-4）：

```
TCD0.CTRLC = TCD_CMPDSEL_bm;
```

图 4-3：CTRLC 寄存器

Bit	7	6	5	4	3	2	1	0
	CMPDSEL	CMPCSEL			FIFTY		AUPDATE	CMPOVR
Access	R/W	R/W			R/W		R/W	R/W
Reset	0	0			0		0	0

图 4-4：全桥输出



3. 频率可以使用以下公式计算：

$$f_{PWM} = \frac{f_{CLK}}{Prescaler_{CNT} \times (CMPASET + CMPACLR + CMPBSET + CMPBCLR + 4)}$$

$$CMPASET + CMPACLR + CMPBSET + CMPBCLR = \frac{f_{CLK}}{f_{Prescaler} \times f_{PWM}} - 4 =$$

$$= \frac{20000000}{4 \times 10000} - 4 = 496 = 0x1F0$$

目标占空比约为 50%，因此 CMPACLR = CMPBCLR。 对于大多数应用，有一个稳定时间。 因此，用户必须考虑到这一点并手动配置外设以包括一点死区时间。 所以对于 0.4 µs 稳定时间，可设置 CMPASET = CMPBSET = 2：

$$CMPACLR = CMPBCLR = \frac{0x1F0 - CMPASET - CMPBSET}{2} = \frac{0x1F0 - 2 - 2}{2} = 0xF6$$

```
TCD0.CMPASET = 0x02;
TCD0.CMPACLR = 0xF6;
TCD0.CMPBSET = 0x02;
TCD0.CMPBCLR = 0xF6;
```

4. 出于故障检测的目的，定时器的输入通道 A 将配置为低电平有效，并且将使能通道的数字滤波器以过滤潜在的尖峰（见图 4-5）。在本例中，通过按下连接到引脚 PC5 的按钮可以从外部触发故障信号：

```
TCD0.EVCTRLA = TCD_CFG_FILTER_gc
               | TCD_EDGE_FALL_LOW_gc
               | TCD_TRIGEI_bm;
```

图 4-5：EVCTRL 寄存器

Bit	7	6	5	4	3	2	1	0
	CFG[1:0]			EDGE		ACTION		TRIGEI
Access	R/W	R/W		R/W		R/W		R/W
Reset	0	0		0		0		0

bits 7:6 CFG[1:0]: Event Configuration bits

Value	Name	Description
0x0	NEITHER	Neither filter nor asynchronous event is enabled.
0x1	FILTERON	Input capture noise cancellation filter enabled.
0x2	ASYNCON	Asynchronous event output qualification enabled.
other	-	Reserved.

5. 通过使用 INPUTCTRLA 寄存器，定时器可以配置为以各种方式响应输入信号。

   在本例中，当在通道 A 上检测到下降沿时，定时器停止并复位。 计数器将在输入信号的下一个上升沿重新启动。

图 4-6：INPUTCTRLA 寄存器

Bit	7	6	5	4	3	2	1	0
					INPUTMODE[3:0]			
Access					R/W	R/W	R/W	R/W
Reset					0	0	0	0

bits 3:0 INPUTMODE[3:0]: Input Mode bits

Value	Name	Description
0x0	NONE	Input has no action
0x1	JMPWAIT	Stop output, jump to opposite compare cycle and wait
0x2	EXECWAIT	Stop output, execute opposite compare cycle and wait
0x3	EXECFAULT	Stop output, execute opposite compare cycle while fault active
0x4	FREQ	Stop all outputs, maintain frequency
0x5	EXECDT	Stop all outputs, execute dead time while fault active
0x6	WAIT	Stop all outputs, jump to next compare cycle and wait
0x7	WAITSW	Stop all outputs, wait for software action
0x8	EDGETRIG	Stop output on edge, jump to next compare cycle
0x9	EDGETRIGFREQ	Stop output on edge, maintain frequency
0xA	LVLTRIGFREQ	Stop output at level, maintain frequency

6. 在从 CTRLA 寄存器启动定时器之前，STATUS 寄存器的 ENRDY 位应为 1。

   此外，从 CTRLA 中选择 20 MHz 时钟和预分频值 4。

```
while(!(TCD0.STATUS & TCD_ENRDY_bm))
{
    ;
}

TCD0.CTRLA = TCD_CLKSEL_20MHZ_gc | TCD_CNTPRES_DIV4_gc | TCD_ENABLE_bm;
```

在此示例中，使用了所有四个输出通道。 前一示例给出了详细过程。

```
void TCD0_enableOutputChannels(void)
{
        CPU_CCP = CCP_IOREG_gc;

        TCD0.FAULTCTRL = TCD_CMPAEN_bm | TCD_CMPBEN_bm | TCD_CMPCEN_bm |
```

```
        TCD_CMPDEN_bm;
}
```

7. 事件系统应配置为将事件发生器链接到 TCD 实例。在本例中，通过按下连接到 PC5 的按钮来模拟故障信号。 初始化代码如下所示，但事件系统配置详细信息不在本文讨论范围内。

```
void EVENT_SYSTEM_init(void)
{
        EVSYS.ASYNCCH2 = EVSYS_ASYNCCH2_PORTC_PIN5_gc;

        EVSYS.ASYNCUSER6 = EVSYS_ASYNCUSER6_ASYNCCH2_gc;
}
```

8. 通道 A和B与PA4 和PA5引脚相连，通道C和D与PC0和PC1引脚相连。 这些引脚必须配置为输出。
   PC5 配置为输入，连接到 ATtiny817 Xplained Mini 用户按钮，并使能内部上拉电阻。
   上拉电阻为引脚提供默认状态“1”。 因此，按钮将用来在按下时将引脚驱动为低电平（逻辑“0”）。

```
PORTA.DIR |= PIN4_bm
           | PIN5_bm;

PORTC.DIR |= PIN0_bm
           | PIN1_bm;

PORTC.DIR &= ~PIN5_bm;

PORTC.PIN5CTRL = PORT_PULLUPEN_bm;
```

该应用将配置 TCD 实例以产生 4 个 PWM 信号，频率为 10 kHz，占空比约为 50%，成对同步。

此外，当输入通道上出现故障信号时，定时器将停止并等待，直到信号变为“安全”状态。

View Code Example on GitHub
Click to browse repository

提示：完整代码示例也可以在附录部分中找到。

5. 参考资料

1. ATtiny817 网页：https://www.microchip.com/wwwproducts/en/ATTINY817
2. ATtiny417/817 – 带独立于内核的外设且采用 picoPower®技术的 AVR®单片机（DS40001901）
3. ATtiny817 Xplained MINI 网页：https://www.microchip.com/DevelopmentTools/ProductDetails/ATTINY817-XMINI

6. 附录

例 6-1：产生互补驱动信号的源代码

```
#define SIGNAL_PERIOD_EXAMPLE_VALUE            (0xC8)
#define SIGNAL_DUTY_CYCLE_EXAMPLE_VALUE        (0x64)

#include <avr/io.h>
/*使用默认时钟 3.3 MHz */

void TCD0_init(void);
void TCD0_enableOutputChannels(void);
void PORT_init(void);

void TCD0_init(void)
{
    /* 设置波形模式 */
    TCD0.CTRLB = TCD_WGMODE_DS_gc;

    /* 设置信号周期 */
    TCD0.CMPBCLR = SIGNAL_PERIOD_EXAMPLE_VALUE;

    /* 信号交替有效，需要一个小的对称死区*/
    TCD0.CMPBSET = SIGNAL_DUTY_CYCLE_EXAMPLE_VALUE
+ 1;
    TCD0.CMPASET = SIGNAL_DUTY_CYCLE_EXAMPLE_VALUE - 1;

    /* 确保 ENRDY 位置 1 */
    while(!(TCD0.STATUS & TCD_ENRDY_bm))
    {
        ;
    }

   TCD0.CTRLA = TCD_CLKSEL_20MHZ_gc        /* 选择定时器的时钟 */
              | TCD_CNTPRES_DIV1_gc        /* 选择预分频值 */
              | TCD_ENABLE_bm;             /* 使能定时器 */

}

void TCD0_enableOutputChannels(void)
{
    /* enable write protected register
    */ CPU_CCP = CCP_IOREG_gc;

    TCD0.FAULTCTRL = TCD_CMPAEN_bm          /* 使能通道 A */
                   | TCD_CMPBEN_bm;         /* 使能通道 B */
}

void PORT_init(void)
{
    PORTA.DIR |= PIN4_bm                    /* 将引脚 4 设置为输出 */
              | PIN5_bm;                    /* 将引脚 5 设置为输出 */
}

int main(void)
{
    PORT_init();

    TCD0_enableOutputChannels();

    TCD0_init();

    /* 此处替换为您的应用程序 */
    while (1)
    {
        ;
    }
}
```

例 6-2：使用输入事件控制同步信号的源代码

```
#define SETTLING_TIME_EXAMPLE_VALUE        (0x02)
#define DUTY CYCLE EXAMPLE VALUE        (0xF6)

#include <avr/io.h>
/*使用默认时钟 3.3 MHz */

void TCD0_init(void);
void TCD0_enableOutputChannels(void);
void EVENT_SYSTEM_init(void);
void PORT_init(void);

void TCD0_init(void)
{
    /* 设置波形模式*/
    TCD0.CTRLB = TCD_WGMODE_FOURRAMP_gc;

    /* 设置通道 D 匹配通道 B */
    TCD0.CTRLC = TCD_CMPDSEL_bm;

    /*设置信号的稳定时间和占空比*/
    TCD0.CMPASET = SETTLING_TIME_EXAMPLE_VALUE;
    TCD0.CMPACLR = DUTY_CYCLE_EXAMPLE_VALUE;
    TCD0.CMPBSET = SETTLING_TIME_EXAMPLE_VALUE;
    TCD0.CMPBCLR = DUTY CYCLE EXAMPLE VALUE;

    TCD0.EVCTRLA = TCD_CFG_FILTER_gc             /* 设置抗尖峰滤波器 */
                 | TCD_EDGE_FALL_LOW_gc          /* 设置默认状态 */
                 | TCD TRIGEI bm;                /* 使能输入通道 A */

    /* 设置输入模式 */
    TCD0.INPUTCTRLA = TCD_INPUTMODE_WAIT_gc;

    /* 确保 ENRDY 位置 1 */
    while(!(TCD0.STATUS & TCD_ENRDY_bm))
    {
        ;
    }

    TCD0.CTRLA = TCD_CLKSEL_20MHZ_gc             /* 选择定时器的时钟 */
               | TCD_CNTPRES_DIV4_gc             /* 选择预分频值 */
               | TCD_ENABLE_bm;                  /* 使能定时器 */
}

void TCD0_enableOutputChannels(void)
{
    /* 使能写保护寄存器 */
    CPU CCP = CCP IOREG gc;

    TCD0.FAULTCTRL = TCD_CMPAEN_bm               /* 使能通道 A */
                   | TCD_CMPBEN_bm               /* 使能通道 B */
                   | TCD_CMPCEN_bm               /* 使能通道 C */
                   | TCD CMPDEN bm;              /* 使能通道 D */
}

void EVENT_SYSTEM_init(void)
{
    EVSYS.ASYNCCH2 = EVSYS_ASYNCCH2_PORTC_PIN5_gc;

    EVSYS.ASYNCUSER6 = EVSYS_ASYNCUSER6_ASYNCCH2_gc;
}

void PORT_init(void)
{
    /* 将端口 A 的引脚 4 和引脚 5 设置为输出*/
    PORTA.DIR |= PIN4_bm
               | PIN5_bm;

    /* 将端口 C 的引脚 0 和引脚 1 设置为输出 */
    PORTC.DIR |= PIN0_bm
               | PIN1_bm;

    /* 将端口 C 的引脚 5 设置为输入*/
    PORTC.DIR &= ~PIN5 bm;
```

```
    /* 使能端口 C 引脚 5 的上拉电阻 */
    PORTC.PIN5CTRL = PORT_PULLUPEN_bm;
}

int main(void)
{
    PORT_init();

    EVENT_SYSTEM_init();

    TCD0_enableOutputChannels();

    TCD0_init();

    /* 此处替换为您的应用程序 */
    while (1)
    {
        ;
    }
}
```

Microchip 网站

Microchip 网站 http://www.microchip.com/为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的互联网浏览器即可访问，网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及归档软件
- **一般技术支持**——常见问题（FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的变更通知客户服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 http://www.microchip.com/。在“支持”（Support）下，点击“变更通知客户”（Customer Change Notification）服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师（FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过以下网站获得技术支持： http://www.microchip.com/support

Microchip 器件代码保护功能

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下，Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿意与关心代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是“牢不可破”的。

代码保护功能处于持续发展中。Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了《数字器件千年版权法案（Digital Millennium Copyright Act）》。如

果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

法律声明

本出版物中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。Microchip 对因这些信息及使用这些信息而引起的后果不承担任何责任。如果将 Microchip 器件用于生命维持和/或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切伤害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任，并加以赔偿。除非另外声明，否则在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

商标

Microchip 的名称和徽标组合、Microchip 徽标、Adaptec、AnyRate、AVR、AVR 徽标、AVR Freaks、BesTime、BitCloud、chipKIT、chipKIT 徽标、CryptoMemory、CryptoRF、dsPIC、FlashFlex、flexPWR、HELDO、IGLOO、JukeBlox、KeeLoq、Kleer、LANCheck、LinkMD、maXStylus、maXTouch、MediaLB、megaAVR、Microsemi、Microsemi 徽标、MOST、MOST 徽标、MPLAB、OptoLyzer、PackeTime、PIC、picoPower、PICSTART、PIC32 徽标、PolarFire、Prochip Designer、QTouch、SAM-BA、SenGenuity、SpyNIC、SST、SST 徽标、SuperFlash、Symmetricom、SyncServer、Tachyon、TempTrackr、TimeSource、tinyAVR、UNI/O、Vectron 及 MEGA 均为 Microchip Technology Inc.在美国和其他国家或地区的注册商标。

APT、ClockWorks、The Embedded Control Solutions Company、EtherSynch、FlashTec、Hyper Speed Control、HyperLight Load、IntelliMOS、Libero、motorBench、mTouch、Powermite 3、PrecisionEdge、ProASIC、ProASIC Plus、ProASIC Plus 徽标、Quiet-Wire、SmartFusion、SyncWorld、Temux、TimeCesium、TimeHub、TimePictra、TimeProvider、Vite、WinPath 和 ZL 均为 Microchip Technology Inc.在美国的注册商标。

Adjacent Key Suppression、AKS、Analog-for-the-Digital Age、Any Capacitor、AnyIn、AnyOut、BlueSky、BodyCom、CodeGuard、CryptoAuthentication、CryptoAutomotive、CryptoCompanion、CryptoController、dsPICDEM、dsPICDEM.net、Dynamic Average Matching、DAM、ECAN、EtherGREEN、In-Circuit Serial Programming、ICSP、INICnet、Inter-Chip Connectivity、JitterBlocker、KleerNet、KleerNet 徽标、memBrain、Mindi、MiWi、MPASM、MPF、MPLAB Certified 徽标、MPLIB、MPLINK、MultiTRAK、NetDetach、Omniscient Code Generation、PICDEM、PICDEM.net、PICkit、PICtail、PowerSmart、PureSilicon、QMatrix、REAL ICE、Ripple Blocker、SAM-ICE、Serial Quad I/O、SMART-I.S.、SQI、SuperSwitcher、SuperSwitcher II、Total Endurance、TSHARC、USBCheck、VariSense、ViewSpan、WiperLock、Wireless DNA 和 ZENA 均为 Microchip Technology Inc.在美国和其他国家或地区的商标。

SQTP 为 Microchip Technology Incorporated 在美国的服务标记。

Adaptec 徽标、Frequency on Demand、Silicon Storage Technology 和 Symmcom 为 Microchip Technology Inc.在除美国外的国家或地区的注册商标。

GestIC 为 Microchip Technology Inc.的子公司 Microchip Technology Germany II GmbH & Co. & KG 在除美国外的国家或地区的注册商标。

在此提及的所有其他商标均为各持有公司所有。

ISBN： 978-1-5224-4327-8

质量管理体系

有关 Microchip 质量管理体系的更多信息，请访问 www.microchip.com/quality。

全球销售及服务网点

美洲	亚太地区	亚太地区	欧洲
公司总部 2355 West Chandler Blvd. 钱德勒，亚利桑那州 85224-6199 电话：480-792-7200 传真：480-792-7277 技术支持： http://www.microchip.com/support 网址： www.microchip.com **亚特兰大** 德卢斯，佐治亚州 电话：678-957-9614 传真：678-957-1455 **奥斯汀，德克萨斯州** 电话：512-257-3370 **波士顿** 韦斯特伯鲁，马萨诸塞州 电话：774-760-0087 传真：774-760-0088 **芝加哥** 艾塔斯卡，伊利诺伊州 电话：630-285-0071 传真：630-285-0075 **达拉斯** 阿迪森，德克萨斯州 电话：972-818-7423 传真：972-818-2924 **底特律** 诺维，密歇根州 电话：248-848-4000 **休斯顿，德克萨斯州** 电话：281-894-5983 **印第安纳波利斯** 诺布尔斯维尔，印第安纳州 电话：317-773-8323 传真：317-773-5453 电话：317-536-2380 **洛杉矶** 米慎维荷，加利福尼亚州 电话：949-462-9523 传真：949-462-9608 电话：951-273-7800 **罗利，北卡罗来纳州** 电话：919-844-7510 **纽约，纽约州** 电话：631-435-6000 **圣何塞，加利福尼亚州** 电话：408-735-9110 电话：408-436-4270 **加拿大 - 多伦多** 电话：905-695-1980 传真：905-695-2078	**澳大利亚 - 悉尼** 电话：61-2-9868-6733 **中国 - 北京** 电话：86-10-8569-7000 **中国 - 成都** 电话：86-28-8665-5511 **中国 - 重庆** 电话：86-23-8980-9588 **中国 - 东莞** 电话：86-769-8702-9880 **中国 - 广州** 电话：86-20-8755-8029 **中国 - 杭州** 电话：86-571-8792-8115 **中国 - 香港特别行政区** 电话：852-2943-5100 **中国 - 南京** 电话：86-25-8473-2460 **中国 - 青岛** 电话：86-532-8502-7355 **中国 - 上海** 电话：86-21-3326-8000 **中国 - 沈阳** 电话：86-24-2334-2829 **中国 - 深圳** 电话：86-755-8864-2200 **中国 - 苏州** 电话：86-186-6233-1526 **中国 - 武汉** 电话：86-27-5980-5300 **中国 - 西安** 电话：86-29-8833-7252 **中国 - 厦门** 电话：86-592-2388138 **中国 - 珠海** 电话：86-756-3210040	**印度 - 班加罗尔** 电话：91-80-3090-4444 **印度 - 新德里** 电话：91-11-4160-8631 **印度 - 浦那** 电话：91-20-4121-0141 **日本 - 大阪** 电话：81-6-6152-7160 **日本 - 东京** 电话：81-3-6880- 3770 **韩国 - 大邱** 电话：82-53-744-4301 **韩国 - 首尔** 电话：82-2-554-7200 **马来西亚 - 吉隆坡** 电话：60-3-7651-7906 **马来西亚 - 槟榔屿** 电话：60-4-227-8870 **菲律宾 - 马尼拉** 电话：63-2-634-9065 **新加坡** 电话：65-6334-8870 **台湾地区 - 新竹** 电话：886-3-577-8366 **台湾地区 - 高雄** 电话：886-7-213-7830 **台湾地区 - 台北** 电话：886-2-2508-8600 **泰国 - 曼谷** 电话：66-2-694-1351 **越南 - 胡志明市** 电话：84-28-5448-2100	**奥地利 - 韦尔斯** 电话：43-7242-2244-39 传真：43-7242-2244-393 **丹麦 - 哥本哈根** 电话：45-4450-2828 传真：45-4485-2829 **芬兰 - 埃斯波** 电话：358-9-4520-820 **法国 - 巴黎** 电话：33-1-69-53-63-20 传真：33-1-69-30-90-79 **德国 - 加兴** 电话：49-8931-9700 **德国 - 哈恩** 电话：49-2129-3766400 **德国 - 海尔布隆** 电话：49-7131-72400 **德国 - 卡尔斯鲁厄** 电话：49-721-625370 **德国 - 慕尼黑** 电话：49-89-627-144-0 传真：49-89-627-144-44 **德国 - 罗森海姆** 电话：49-8031-354-560 **以色列 - 若那那市** 电话：972-9-744-7705 **意大利 - 米兰** 电话：39-0331-742611 传真：39-0331-466781 **意大利 - 帕多瓦** 电话：39-049-7625286 **荷兰 - 德卢内市** 电话：31-416-690399 传真：31-416-690340 **挪威 - 特隆赫姆** 电话：47-72884388 **波兰 - 华沙** 电话：48-22-3325737 **罗马尼亚 - 布加勒斯特** 电话：40-21-407-87-50 **西班牙 - 马德里** 电话：34-91-708-08-90 传真：34-91-708-08-91 **瑞典 - 哥德堡** 电话：46-31-704-60-40 **瑞典 - 斯德哥尔摩** 电话：46-8-5090-4654 **英国 - 沃金厄姆** 电话：44-118-921-5800 传真：44-118-921-5820