

在 Kinetis 上使用 DMA 进行脉冲计数

1 简介

本应用笔记阐述了如何使用 DMA 和通用 IO 模块进行脉冲计数。许多 Kinetis 系列微控制器具有低功耗定时器 (LPT) 和 FlexTimer，可用于脉冲输入捕捉。然而，当应用程序需要对多个脉冲输入进行计数时，低功耗定时器则无法满足需求。例如，LPTMR (低功耗定时器模块) 内部只有一个计数器。该计数器只能用作单脉冲计数器。

本应用程序旨在使用通用 IO 来捕捉多脉冲输入，并且使用 DMA 模块通道倒计数寄存器来进行脉冲计数。

通过使用该方法，在无 CPU 干预的情况下，Kinetis 芯片最多可同时对 5 个通道的脉冲输入进行计数。

由于 Kinetis MCU 的兼容性，不同的 Kinetis 子系列 (K60 除外) 芯片上均可使用该应用程序。所有 Kinetis K 系列共用相同的模块 (IP blocks)，因此在整个 K 子系列中实现代码复用比较简单。

2 功能说明

内容

1	简介.....	1
2	功能说明.....	1
3	初始化和应用程序.....	2
4	结论.....	4
5	功能限制.....	5
5.1	最大 DMA 速率.....	5
5.2	DMA 主循环计数器.....	5

当外设请求使能并且触发信号到来时，会产生一个 DMA 请求；当 DMA 完成一次循环传输后，DMA 通道计数器会从重载值开始向下计数，直到为零为止。如果将 DMA 主循环计数器设置为一个适当的值，并且将 DMA 触发源设置为来自 GPIO 的外部边沿捕捉，我们可以通过读取 DMA 通道的向下计数寄存器，在该端口上进行脉冲输入捕捉。

Kinetis K 系列集成了一个 DMA 请求多路复用器，最多可允许 64 个 DMA 请求信号映射至任何一个 DMA 通道。因此，我们安排 PORTA-PORTE 的上升/下降触发源到 DMA 通道 0-5。

表 1. 通道请求源

DMA 通道编号	DMA 请求源编号	源描述
DMA 通道 0	49	PORTA 请求
DMA 通道 1	50	PORTB 请求
DMA 通道 2	51	PORTC 请求
DMA 通道 3	52	PORTD 请求
DMA 通道 4	53	PORTE 请求

以下代码片段显示了将 DMA 通道请求源配置为相应的 PORT 请求。

```
DMAMUX0_CHCFG0 = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(49); //PORTA request
DMAMUX0_CHCFG1 = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(50); //PORTB request
DMAMUX0_CHCFG2 = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(51); //PORTC request
DMAMUX0_CHCFG3 = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(52); //PORTD request
DMAMUX0_CHCFG4 = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(53); //PORTE request
```

注

由于 DMA 触发源基于 PORT，一个 PORT 实例只能有一个引脚用作 DMA 脉冲输入。因此，脉冲输入的最大通道数与芯片的 PORT 模块实例数相等。

3 初始化和应用程序

若要初始化 DMA 多脉冲计数功能，需执行以下步骤：

对于 DMA 模块：

1. 使能 DMAMUX 和 DMA 时钟，配置 DMAMUX 请求源。
2. 将 DMA 源和目标地址设为虚拟变量地址。
3. 将 DMA 源和目标的主/次地址的调整值和偏移量设为零。
4. 将每个 DMA 通道的向下计数计数器（CITER 和 BITER）设为相应值。
5. 使能 DMA 通道请求等待外部触发器的信号。

对于通用 IO 和 PORT 模块：

1. 设置 PORT 引脚对通用 IO 复用。
2. 在 PORT_MUX 寄存器上使能 DMA 的上升沿/下降沿请求。

对于应用程序：

1. 初始化 DMA 和通用 IO 模块。
2. 在应用程序中读取 DMA 通道计数器以获取脉冲计数值。

示例代码如下所示：

```
/* clock gate initialization */
SIM->SCGC5 |= SIM_SCGC5_PORTA_MASK,
SIM->SCGC5 |= SIM_SCGC5_PORTB_MASK,
SIM->SCGC5 |= SIM_SCGC5_PORTC_MASK,
SIM->SCGC5 |= SIM_SCGC5_PORTD_MASK,
```

```

SIM->SCGC5 |= SIM_SCGC5_PORTE_MASK,

/* set pin to GPIO function and enable DMA request source */
PORTA->PCR[0] |= PORT_PCR_MUX(1) | PORT_PCR_IRQC(1);
PORTA->PCR[0] |= PORT_PCR_MUX(1) | PORT_PCR_IRQC(1);
PORTA->PCR[0] |= PORT_PCR_MUX(1) | PORT_PCR_IRQC(1);
PORTA->PCR[0] |= PORT_PCR_MUX(1) | PORT_PCR_IRQC(1);
PORTA->PCR[0] |= PORT_PCR_MUX(1) | PORT_PCR_IRQC(1);

/* enable DMA and DMAMUX clock */
SIM->SCGC6 |= SIM_SCGC6_DMAMUX_MASK;
SIM->SCGC7 |= SIM_SCGC7_DMA_MASK;

```

在配置 DMA MUX 触发器和源数量之前，禁用 DMAMUX->CFG[ENBL]位。

```

/* clears register for changing source and trigger */
DMAMUX->CHCFG[1] = 0;
DMAMUX->CHCFG[2] = 0;
DMAMUX->CHCFG[3] = 0;
DMAMUX->CHCFG[4] = 0;

/* set DMA channel request source */
DMAMUX->CHCFG[0] = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(49);
DMAMUX->CHCFG[1] = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(50);
DMAMUX->CHCFG[2] = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(51);
DMAMUX->CHCFG[3] = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(52);
DMAMUX->CHCFG[4] = DMA_MUX_CHCFG_ENBL_MASK | DMA_MUX_CHCFG_SOURCE(53);

```

在初始化 DMA 寄存器之前，需要向 DMA 的源和目标地址寄存器填入一个无用变量的地址值。

DMA 模块初始化。例如，仅使用 DMA 通道 0。

```

/* source configuration */
DMA0->TCD[0].SADDR = &dummy;
DMA0->TCD[0].ATTR = DMA_ATTR_SSIZE(0);
DMA0->TCD[0].SOFF= 0; /* no address shift after each transfer */
DMA0->TCD[0].SLAST = 0;

/* destination configuration */
DMA0->TCD[0].DADDR = &dummy;
DMA0->TCD[0].ATTR = DMA_ATTR_DSIZE(0);
DMA0->TCD[0].DOFF= 0;
DMA0->TCD[0].DLAST_SGA= 0;

/* set CITER and BITER to maximum value */
DMA0->TCD[0].CITER_ELINKNO = DMA_CITER_ELINKNO_CITER_MASK;
DMA0->TCD[0].BITER_ELINKNO = DMA_CITER_ELINKNO_BITER_MASK;
DMA0->TCD[0].NBYTES_MLNO = 1; /* transfer one byte on each trigger arrived */

/* enable auto close request */
DMA0->TCD[0].CSR |= DMA_CSR_DREQ_MASK;

/* start transfer */
DMA0->SERQ = DMA_SERQ_SERQ(0);

```

在应用程序中读取 DMA 通道计数器。脉冲计数值为 BITER 减 CITER 的值。

注

主迭代计数耗尽（即 CITER 达到零）之前，可在任意时刻从应用程序中读取 DMA 通道计数器。一旦 CITER 达到零，用户需要将 CITER 寄存器的值重置为 BITER，然后重新开始计数操作。

```

uint32_t value;
value = DMA0->TCD[0].BITER_ELINKNO - DMA0->TCD[0].CITER_ELINKNO;

```

4 结论

DMA 多脉冲计数器完全由内部硬件逻辑实现。用户只需要读取 DMA 通道向下计数寄存器以获取脉冲计数值。整个过程不占用任何 CPU 资源。

图 1 显示了使用 DMA 捕捉 3 种不同频率（10 KHz、20 KHz 和 1 KHz）方波的结果。输入方波：

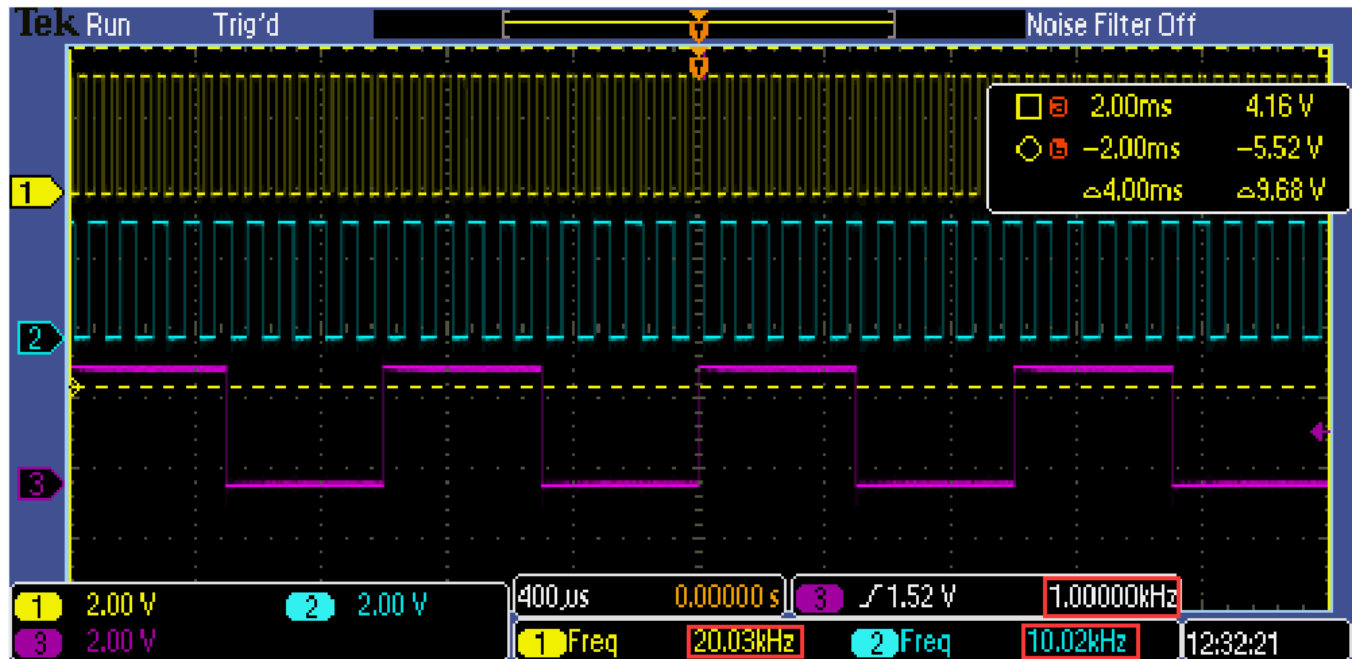
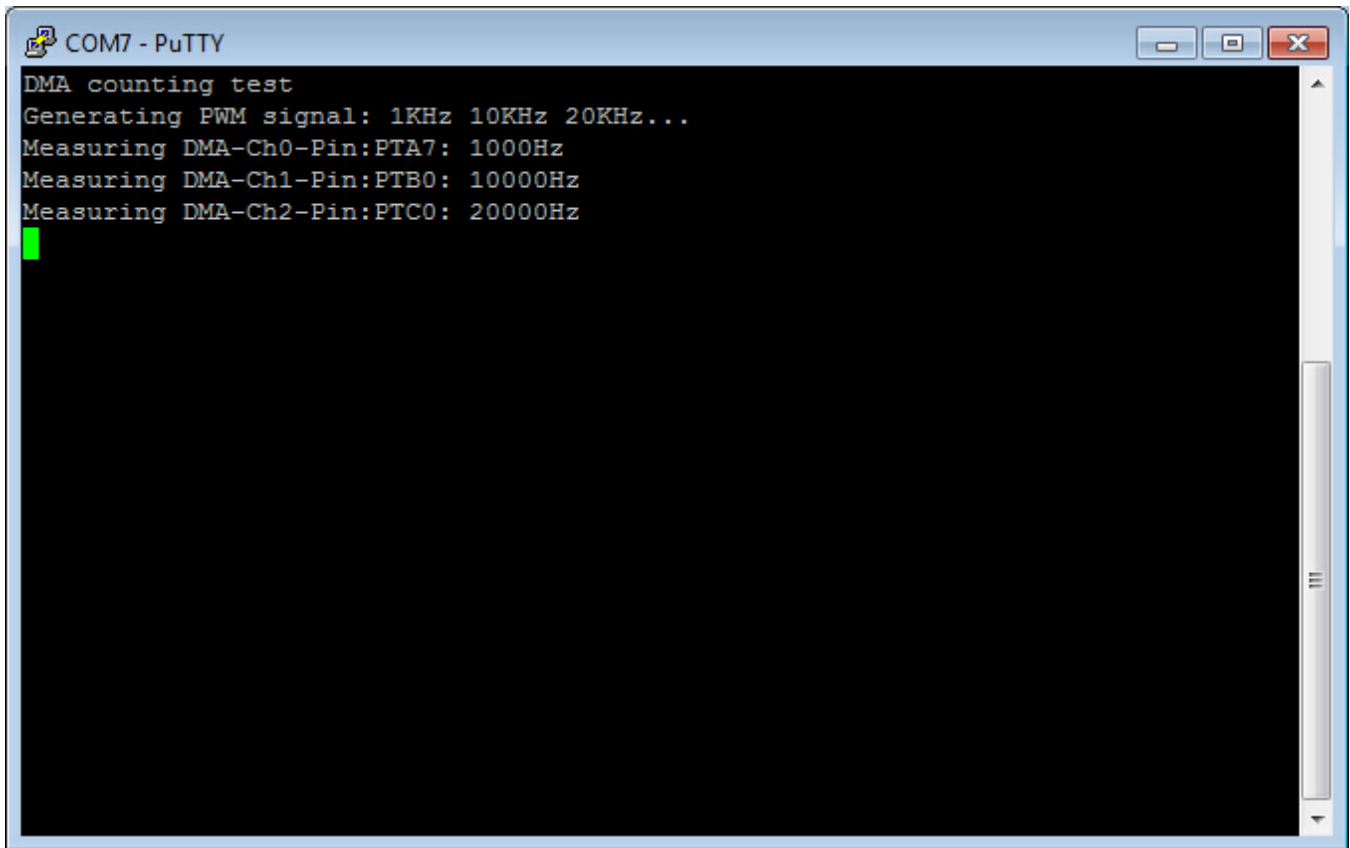


图 1. 输入方波

若要测量方波频率：

1. 使能 PIT 定时器以产生中断。
2. 在中断服务程序中读取每个 DMA 通道计数器，并复位计数器。
3. 在串行终端上显示计数结果。

A screenshot of a PuTTY terminal window titled 'COM7 - PuTTY'. The terminal displays the following text: 'DMA counting test', 'Generating PWM signal: 1KHz 10KHz 20KHz...', 'Measuring DMA-Ch0-Pin:PTA7: 1000Hz', 'Measuring DMA-Ch1-Pin:PTB0: 10000Hz', and 'Measuring DMA-Ch2-Pin:PTC0: 20000Hz'. A green cursor is visible on the line following the last measurement. The terminal window has standard Windows-style window controls (minimize, maximize, close) in the top right corner and a scrollbar on the right side.

```
COM7 - PuTTY
DMA counting test
Generating PWM signal: 1KHz 10KHz 20KHz...
Measuring DMA-Ch0-Pin:PTA7: 1000Hz
Measuring DMA-Ch1-Pin:PTB0: 10000Hz
Measuring DMA-Ch2-Pin:PTC0: 20000Hz
█
```

图 2. 串行终端上的计数结果

5 功能限制

在实现过程中必须考虑几种限制。本文档中针对某些限制提供了解决方案。

5.1 最大 DMA 速率

用于 DMA 硬件请求（特别是单字节传输）的最大传输速率受到一些因素的限制。对于 100 MHz 的器件，可实现的最大速率为 8.7 MHz，因此其最大输入频率不能超过该速率。

5.2 DMA 主循环计数器

在不使用 DMA 通道链接特性的情况下，DMA 寄存器 DMA_TCD_n_CITER_ELINKNO 的主循环计数器限为 15 位（请参见参考手册中的 DMA 章节）。因此，主循环计数限为 32K。用户必须检查 CITER 寄存器并确保其未达到零。如果 CITER 达到零，则 BITER 将重载 BITER 寄存器中的数值。

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

本文档中的信息仅供系统和软件实施方使用 Freescale 产品。本文并未明示或者暗示授予利用本文档信息进行设计或者加工集成电路的版权许可。Freescale 保留对此处任何产品进行更改的权利，恕不另行通知。

Freescale 对其产品在任何特定用途方面的适用性不做任何担保、表示或保证，也不承担因为应用程序或者使用产品或电路所产生的任何责任，明确拒绝承担包括但不限于后果性的或附带性的损害在内的所有责任。Freescale 的数据表和/或规格中所提供的“典型”参数在不同应用中可能并且确实不同，实际性能会随时间而有所变化。所有运行参数，包括“经典值”在内，必须经由客户的技术专家对每个客户的应用程序进行验证。Freescale 未转让与其专利权及其他权利相关的许可。Freescale 销售产品时遵循以下网址中包含的标准销售条款和条件：freescale.com/SalesTermsandConditions。

Freescale, Kinetis, and the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners.

© 2015 Freescale Semiconductor, Inc.

© 2015 飞思卡尔半导体有限公司