AVR®单片机上的独立于内核的外设入门指南

简介

独立于内核的外设（Core Independent Peripheral，CIP）是许多 AVR®器件上都会配备的一类外设。本应用笔记将重点介绍 tinyAVR® 1 系列，其中的一般原理适用于所有配备 CIP 的器件，只有在外设特性和设计方面存在一些差异。

CIP 旨在通过一个或多个外设来处理相应的任务，无需代码或 CPU 监控即可维持正常运行。这样做有很多好处，例如在外设之间提供短暂且可预测的响应时间、降低软件复杂度、缩短软件执行时间以及降低功耗。

tinyAVR® 1 系列器件上配有大量的 CIP。例如：事件系统（EVSYS）、可配置定制逻辑（Configurable Custom Logic，CCL）、定时器/计数器 A 和 B（TCA/TCB）、实时定时器计数器（Real Timer Counter，RTC）、模数转换器（Analog-to-Digital Converter，ADC）和 CRCSCAN。

本应用笔记首先将介绍独立于内核的应用中两个功能最强大的模块：CCL 和事件系统。接着，将通过一个应用示例来介绍如何结合使用 CCL、事件系统、RTC 和 ADC 来对按钮信号进行滤波并独立启动 ADC 转换内核。这对于想要使用 CIP 构建自定义项目的用户来说很有帮助。

特性

目录

1. 相关器件

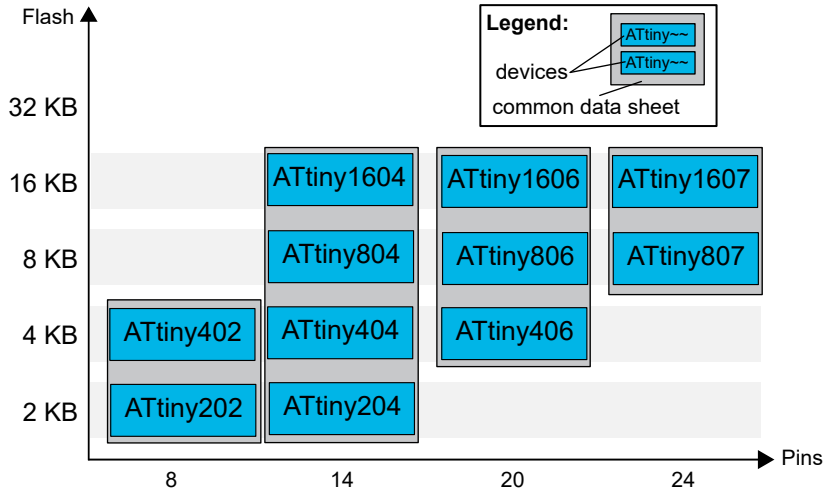
本章列出了文中涉及的相关器件。

1.1 tinyAVR 0 系列

下图所示为 tinyAVR 0 系列器件，注明了不同的引脚数与存储器大小：

- 在垂直方向上，无需修改代码即可实现移植，因为这些器件的引脚和功能完全兼容。
- 水平向左移植会减少引脚数，进而减少可用的功能。

图 1-1. tinyAVR® 0 系列概览



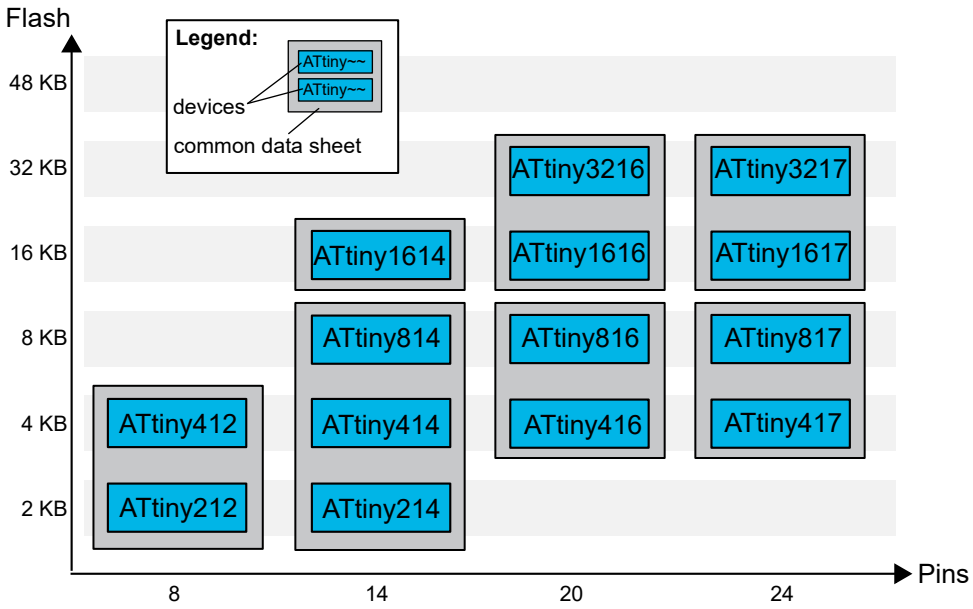
具有不同闪存大小的器件通常也具有不同的 SRAM 和 EEPROM。

1.2 tinyAVR 1 系列

下图所示为 tinyAVR 1 系列器件，注明了不同的引脚数与存储器大小：

- 垂直向上移植无需修改代码，因为这些器件引脚兼容并可提供相同或更多的功能。而向下移植可能需要修改代码，因为某些外设的可用实例数减少。
- 水平向左移植会减少引脚数，进而减少可用的功能。

图 1-2. tinyAVR® 1 系列概览



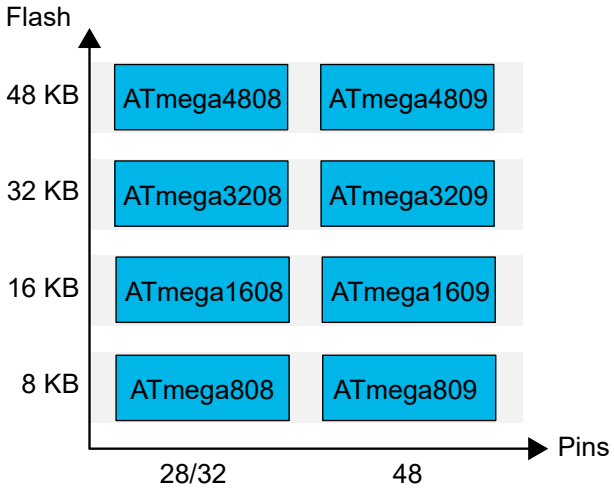
具有不同闪存大小的器件通常也具有不同的 SRAM 和 EEPROM。

1.3 megaAVR 0 系列

下图所示为 megaAVR 0 系列器件，注明了不同的引脚数与存储器大小：

- 无需修改代码即可实现垂直移植，因为这些器件的引脚和功能完全兼容。
- 水平向左移植会减少引脚数，进而减少可用的功能。

图 1-3. megaAVR® 0 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。

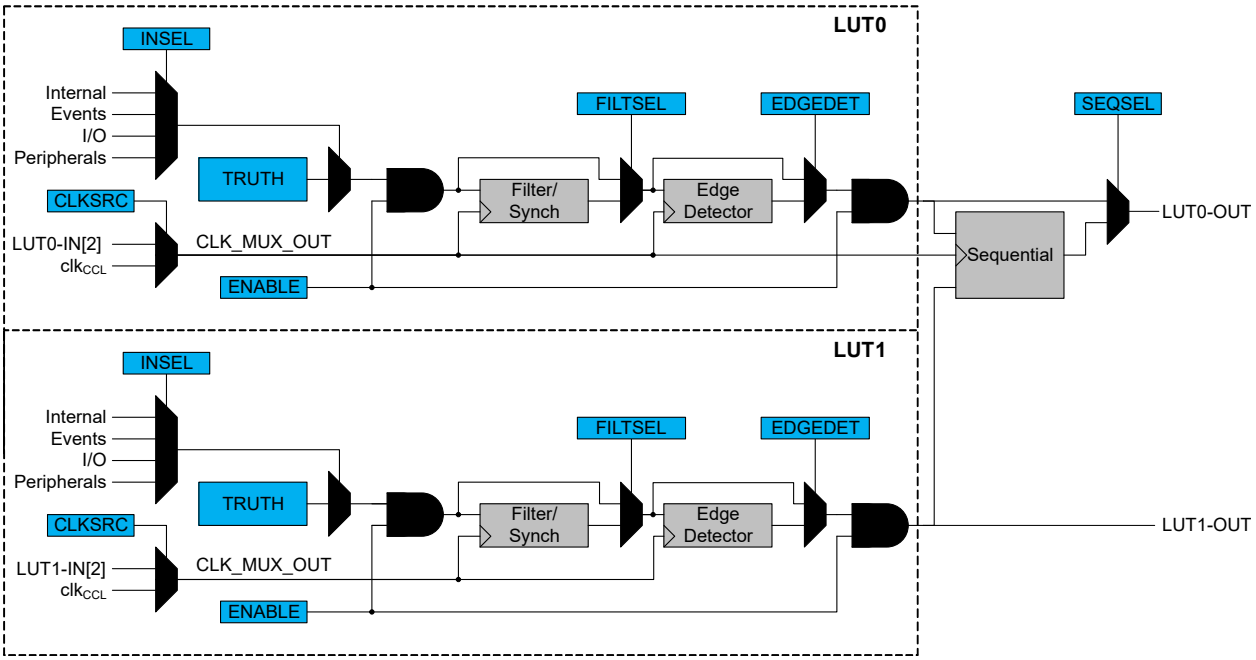
2. CCL 简介

可配置定制逻辑（CCL）是一种可编程逻辑外设，可以连接到各种内部和外部输入，例如器件引脚、事件或其他内部外设。CCL 可以用作器件外设与外部器件之间的“胶连逻辑”。

CCL 外设具有一对查找表（LookUp Table，LUT）。每个 LUT 由三个输入、一个真值表、一个同步器、一个滤波器和一个边沿检测器组成。每个 LUT 均可生成一个输出（具有三个输入的用户可编程逻辑表达式），任何配备 CCL 的器件都至少有两个 LUT。输入可以单独屏蔽。输出可以由各个输入组合生成，并且可进行滤波以消除尖峰。可使能可选的顺序逻辑模块。顺序模块的输入由相邻的两个独立 LUT（LUT0/LUT1）输出分别控制，可生成复杂的波形。

使用 CCL 时无需其他外部逻辑元件，并且能够协助内核处理应用中的时间关键部分。

图 2-1. CCL 概览



2.1 真值表

通过使用 LUT 中的真值表，可以生成最多三个输入的任何逻辑表达式。输入可单独：

- 屏蔽
- 连接到 I/O
- 由外设驱动：
  - 模拟比较器（AC）输出
  - 定时器/计数器（TC）波形输出
  - USART
  - SPI
- 由来自事件系统的内部事件驱动
- 由其他 CCL 子模块驱动

要使 CCL 按预期工作，必须要了解如何通过真值表生成所需的逻辑表达式。

表中的每一个 TRUTH[x]行都将创建一个 3 输入逻辑门，通过选择表中的多个 TRUTH，还可以创建复杂的逻辑表达式。输入位（IN[2:0]）的每一种组合都对应于 TRUTHn 寄存器中的一个位。

表 2-1. LUT 真值表

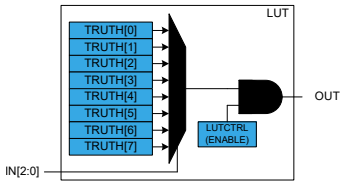
IN[2]	IN[1]	IN[0]	OUT
0	0	0	TRUTH[0]
0	0	1	TRUTH[1]
0	1	0	TRUTH[2]
0	1	1	TRUTH[3]
1	0	0	TRUTH[4]
1	0	1	TRUTH[5]
1	1	0	TRUTH[6]
1	1	1	TRUTH[7]

表 2-2. 可实现的逻辑模块

IN[]	TRUTH	AND	NAND	OR	NOR	XOR	XNOR	NOT
000	TRUTH[0]	0	1	0	1	0	1	1
001	TRUTH[1]	0	1	1	0	1	0	x
010	TRUTH[2]	0	1	1	0	1	0	x
011	TRUTH[3]	0	1	1	0	0	1	x
100	TRUTH[4]	0	1	1	0	1	0	x
101	TRUTH[5]	0	1	1	0	0	1	x
110	TRUTH[6]	0	1	1	0	0	1	x
111	TRUTH[7]	1	0	1	0	1	0	0
		0x80	0x7F	0xFE	0x01	0x96	0x69	0x01

所选的各个 TRUTH[x]将一起进行逻辑或运算，以创建最终的逻辑表达式。

图 2-2.



2.1.1 创建简单的逻辑模块

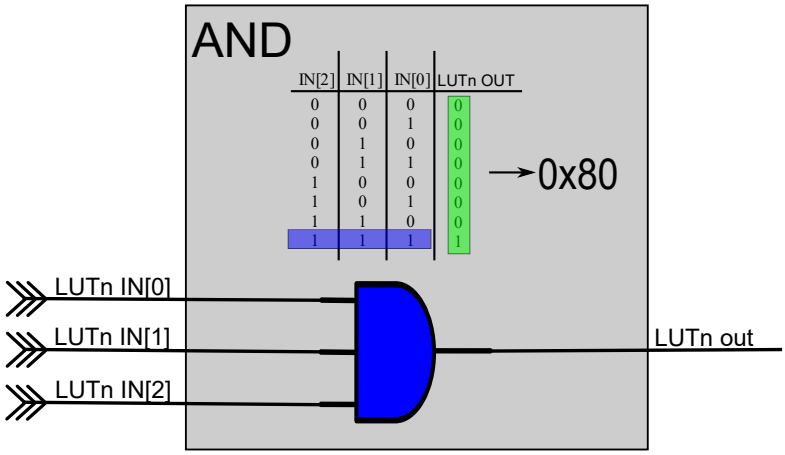
在每个 LUT 上，可以使用最多三个输入的真值表来创建简单的逻辑模块，例如 AND、OR、NAND、NOR 和 XOR。

下面将通过一些示例来说明如何使用三个输入创建各种最常见的逻辑门。

2.1.1.1 “与”门

要使“与”门的输出为高电平（1），必须所有输入均为高电平（1）。通过查看真值表可知，如果使用全部三个输入，则只有 TRUTH[7]满足此要求。这意味着 TRUTH[7]必须为高电平（1），其余项必须为低电平（0），也就是将十六进制值 0x80 置于 TRUTHn 寄存器中。

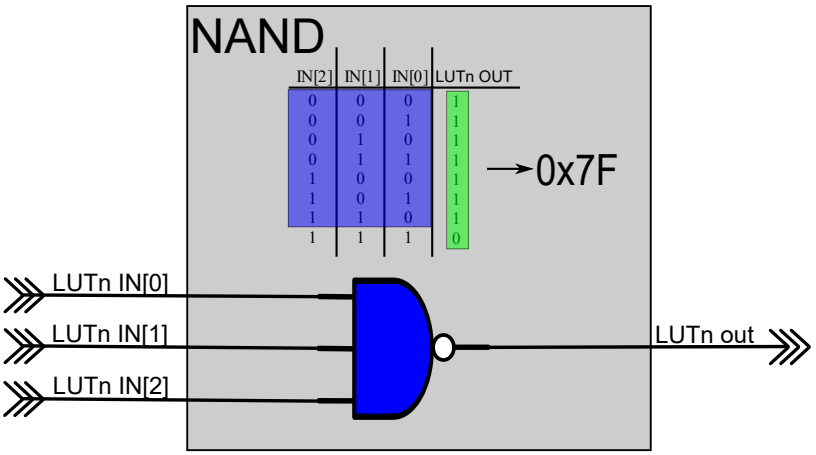
图 2-3. “与”门



2.1.1.2 “与非”门

要使“与非”门的输出为高电平（1），必须至少有一个输入为低电平（0）。如果所有输入均为高电平（1），则输出将为低电平（0）。通过查看真值表可知，除 TRUTH[7]外的所有项均满足此要求。这意味着 TRUTH[0]至 TRUTH[6]必须为高电平（1），TRUTH[7]必须为低电平（0），也就是将十六进制值 0x7F 置于 TRUTHn 寄存器中。

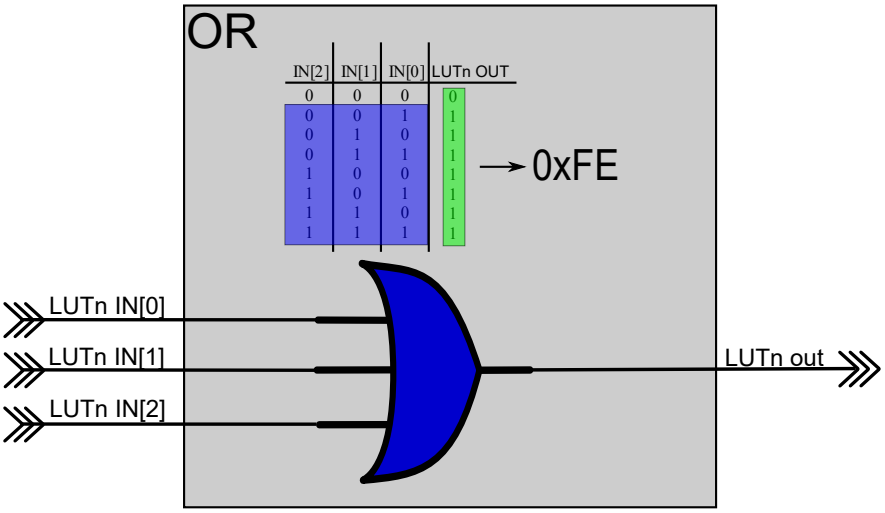
图 2-4. “与非”门



2.1.1.3 “或”门

要使“或”门的输出为高电平（1），必须至少有一个输入为高电平（1）。如果所有输入均为低电平（0），则输出将为低电平（0）。通过查看真值表可知，除 TRUTH[0]外的所有项均满足此要求。这意味着 TRUTH[1]至 TRUTH[7]必须为高电平（1），TRUTH[0]必须为低电平（0），也就是将十六进制值 0xFE 置于 TRUTHn 寄存器中。

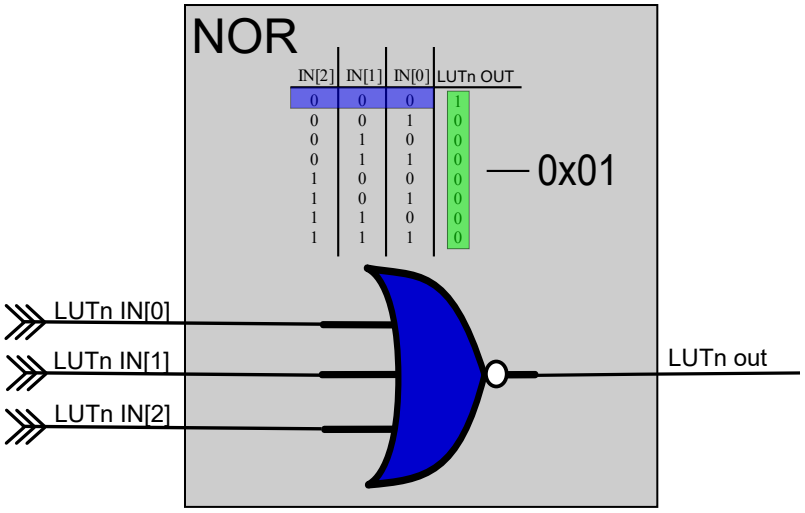
图 2-5. “或”门



2.1.1.4 “或非”门

要使“或非”门的输出为高电平（1），必须所有输入均为低电平（0）。如果任一输入为高电平（1），则输出将为低电平（0）。通过查看真值表可知，只有 TRUTH[0]满足此要求。这意味着 TRUTH[1]至 TRUTH[7]必须为低电平（0），TRUTH[0]必须为高电平（1），也就是将十六进制值 0x01 置于 TRUTHn 寄存器中。

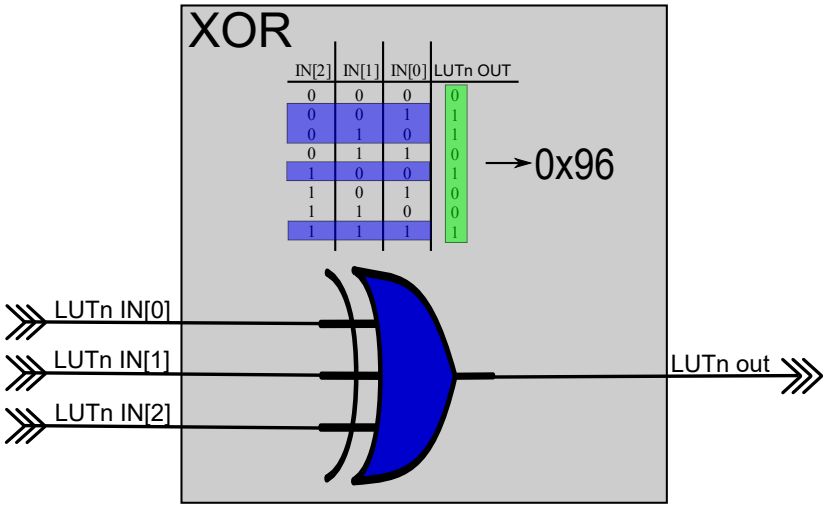
图 2-6. “或非”门



2.1.1.5 “异或”门

要使“异或”门的输出为高电平（1），必须有奇数个输入为高电平（1）。通过查看真值表可知，TRUTH[1]、TRUTH[2]、TRUTH[4]和 TRUTH[7]满足此要求。这意味着这些项必须为高电平（1），其余项必须为低电平（0），也就是将十六进制值 0x96 置于 TRUTHn 寄存器中。

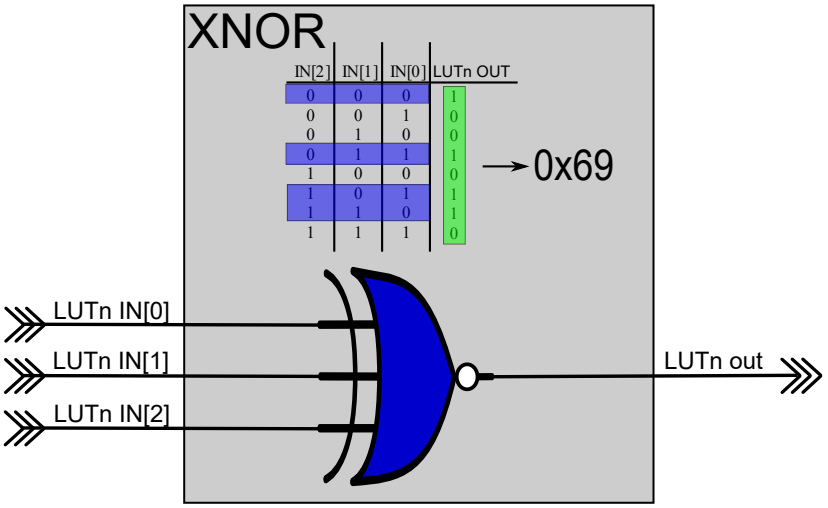
图 2-7. “异或”门



2.1.1.6 “异或非”门

要使“异或非”门的输出为高电平（1），必须有奇数个输入为低电平（0）。通过查看真值表可知，TRUTH[0]、TRUTH[3]、TRUTH[5]和 TRUTH[6]满足此要求。这意味着这些项必须为高电平（1），其余项必须为低电平（0），也就是将十六进制值 0x69 置于 TRUTHn 寄存器中。

图 2-8. “异或非”门



2.1.2 屏蔽输入

每个 LUT 有三个输入可供使用。如果不需要使用全部三个输入，可以屏蔽未使用的输入（保持低电平）。当查看真值表以确定如何设置各个位，从而实现所需的逻辑时，只能使用屏蔽输入为 0 的 TRUTH 位。

屏蔽一个输入时，真值表可以简化为只有两个输入；屏蔽两个输入时，真值表可以简化为只有一个输入。

下表所示为屏蔽 IN[0]时的真值表示例。

表 2-3. 屏蔽 IN[0]时的 LUT 真值表

IN[2]	IN[1]	OUT
0	0	TRUTH[0]
0	1	TRUTH[2]

.....（续）		
IN[2]	**IN[1]**	**OUT**
1	0	TRUTH[4]
1	1	TRUTH[6]

下表所示为屏蔽 IN[1]时的真值表示例。

表 2-4. 屏蔽 IN[1]时的 LUT 真值表

IN[2]	IN[0]	OUT
0	0	TRUTH[0]
0	1	TRUTH[1]
1	0	TRUTH[4]
1	1	TRUTH[5]

下表所示为屏蔽 IN[2]时的真值表示例。

表 2-5. 屏蔽 IN[2]时的 LUT 真值表

IN[1]	IN[0]	OUT
0	0	TRUTH[0]
0	1	TRUTH[1]
1	0	TRUTH[2]
1	1	TRUTH[3]

下表所示为屏蔽 IN[0]和 IN[1]时的真值表示例。

表 2-6. 屏蔽 IN[0]和 IN[1]时的 LUT 真值表

IN[2]	OUT
0	TRUTH[0]
1	TRUTH[4]

下表所示为屏蔽 IN[0]和 IN[2]时的真值表示例。

表 2-7. 屏蔽 IN[0]和 IN[2]时的 LUT 真值表

IN[1]	OUT
0	TRUTH[0]
1	TRUTH[2]

下表所示为屏蔽 IN[1]和 IN[2]时的真值表示例。

表 2-8. 屏蔽 IN[1]和 IN[2]时的 LUT 真值表

IN[0]	OUT
0	TRUTH[0]
1	TRUTH[1]

下面给出了屏蔽各个输入时的示例。

图 2-9. 双输入“与”门，屏蔽 IN[0]

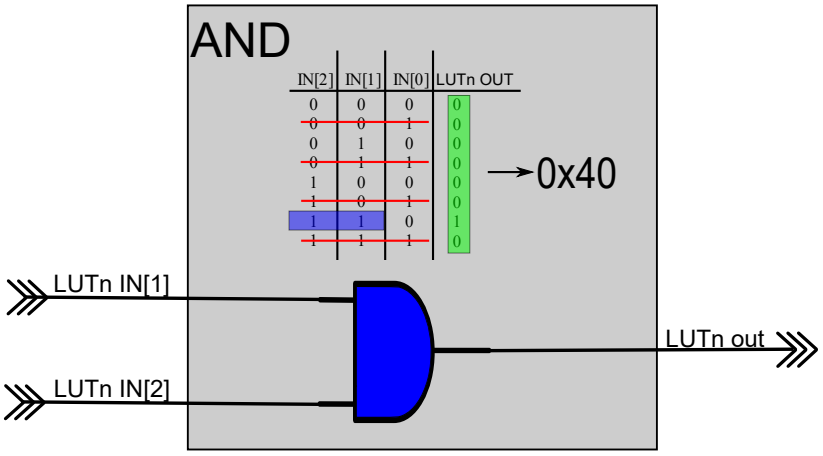


图 2-10. 双输入“或”门，屏蔽 IN[1]

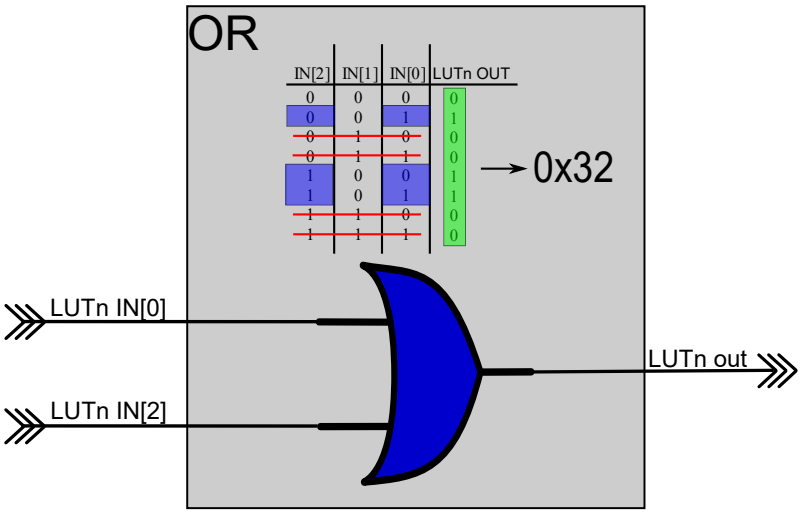
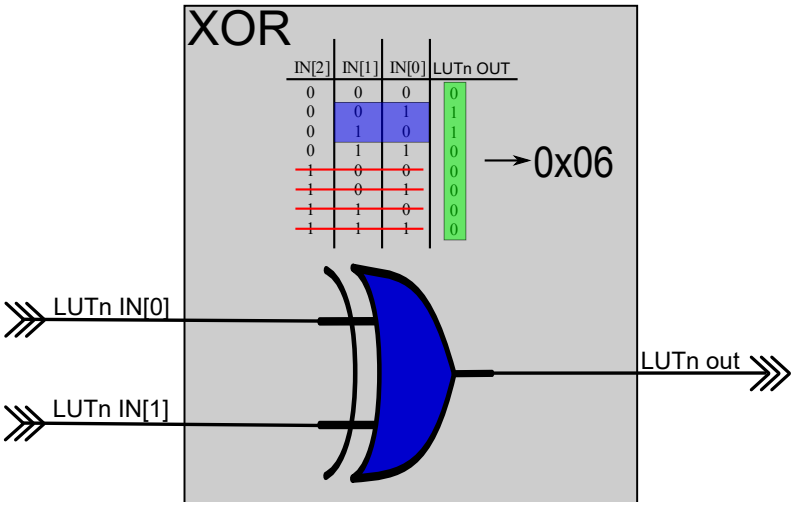


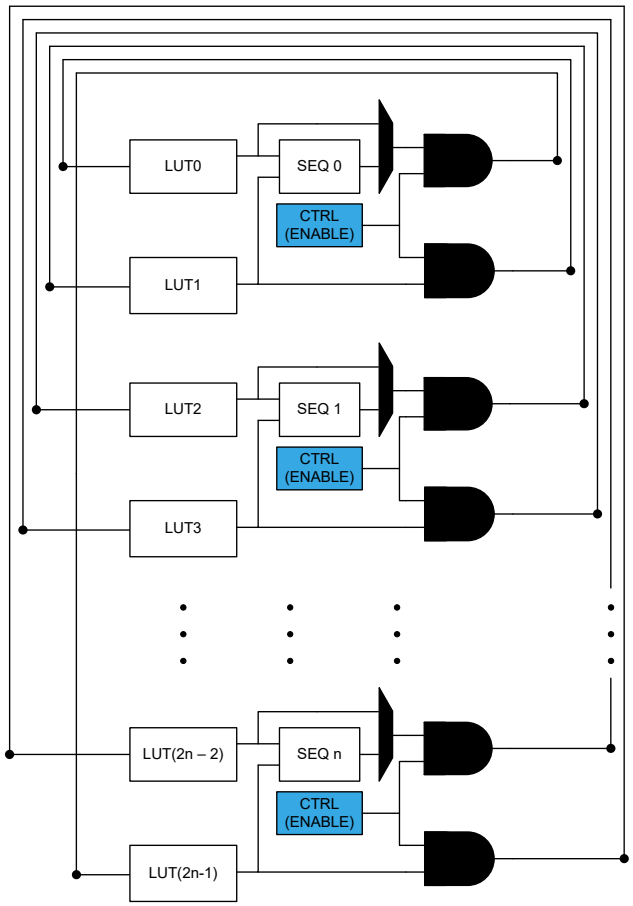
图 2-11. 双输入“异或”门，屏蔽 IN[2]



2.1.3 连接 LUT

连接 LUT 是指将一个 LUT 的输出用作另一个 LUT 的输入。这样一来，可以使用两个 LUT 求解最多五个输入的逻辑表达式。LUTn 只能连接到 LUTn+1，最后一个 LUT 可以连接到第一个 LUT。LUTn 的 LUT 输出可以连接到 LUTn+1 的任一输入。这两个 LUT 可单独创建真值表（即，将二者视为未连接），以此来确定二者分别需要在 TRUTH 寄存器中写入哪些值。

图 2-12. 连接 LUT



在 Atmel Start 中，可以找到使用 CCL 和连接 LUT 的应用笔记和代码示例。

- Quadrature Decoding using CCL with TCA and TCB

2.1.4 如何实现逻辑表达式

使用真值表创建的简单逻辑门可以解决许多任务，但很多时候需要更为复杂的特定逻辑功能。下面将通过一些示例来说明如何基于最多三个输入的真值表来实现逻辑表达式，以及如何使用一个 LUT 来求解这些逻辑表达式。此外还将说明如何通过将两个 LUT 连接在一起来求解最多五个输入的逻辑表达式。

2.1.4.1 使用一个 LUT 实现逻辑表达式

假定要实现以下逻辑表达式：$(A \bullet B) \oplus (B \bullet C)$。

通过该表达式可得出以下真值表：

表 2-9. LUT 真值表

C	B	A	OUT
0	0	0	0
0	0	1	1

.....（续）

C	B	A	OUT
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

通过查看上述真值表可知，需要写入 TRUTH 寄存器的值为 0x72。

2.1.4.2 使用连接的 LUT 实现逻辑表达式

下面将通过示例来说明如何将 LUT0 连接到 LUT1 以及如何填写这两个 LUT 的真值表。

假定要实现以下逻辑表达式：$(A \bullet B \oplus C) + (D \bullet \overline{E})$。

首先，可以创建 LUT0 的真值表。LUT0 将处理逻辑表达式的第一部分 $(A \bullet B \oplus C)$。

通过该表达式可得出以下真值表：

表 2-10. LUT0 真值表

C	B	A	OUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

根据上面的真值表，可通过将 0x60 写入 LUT0 TRUTH 寄存器来实现逻辑表达式的第一部分。

接着来创建 LUT1 的真值表。在此之前，必须确定要使用 LUT1 上的哪个输入。所有输入均可使用，在本示例中，LUT0 输出连接到 LUT1 输入 1。该输入对应真值表的第二列。如果使用输入 0，则对应第一列；如果使用输入 2，则对应第三列。

为了便于创建 LUT1 真值表，可以对表达式进行简化，因为 LUT0 已处理第一部分。创建 LUT1 的真值表时，表达式可简化为以下形式：$(X + D \bullet \overline{E})$，其中 X = $(A \bullet B \oplus C)$

表 2-11. LUT1 真值表

E	X	D	OUT
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0

.....（续）

E	X	D	OUT
1	0	1	0
1	1	0	1
1	1	1	1

根据上面的真值表，可通过将 0xCE 写入 LUT1 TRUTH 寄存器来实现逻辑表达式的第二部分。

2.2 两级同步器、滤波器和边沿检测器

真值表输出是多个输入的组合函数。当输入值发生更改时，可能会导致出现一些短暂的毛刺。这些毛刺可能并不会引起任何问题，但是如果将 LUT 输出设置为触发一个事件（例如，用作定时器输入），则意外出现的毛刺可能会触发意外事件和外设动作。用户可通过滤波器定时消除这些毛刺，从而获得预期的输出。

2.2.1 两级同步器

在同步器选项中，真值表的输出信号将采用两级同步器处理后的时钟。使用该选项时，该信号将最多延迟两个时钟周期。只要时钟上升沿上不存在毛刺，就会使用同步器滤除 LUT 中短于一个时钟周期的毛刺。尽管两级同步器在许多情况下很有用，但仍存在局限性。如果毛刺出现在同步器第一级的上升沿上，则会将其锁存。当退出同步器时，毛刺的时长将变为一个时钟周期。

图 2-13. 两级同步器

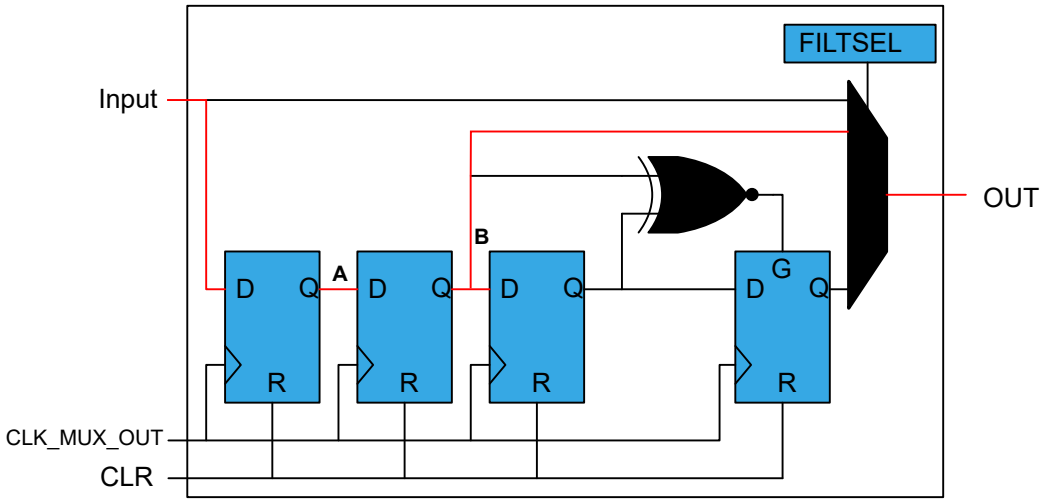
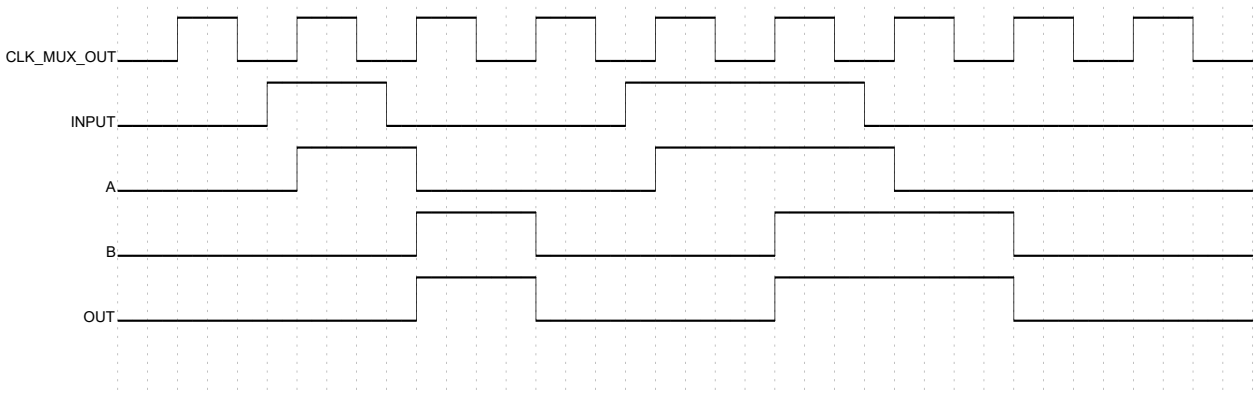


图 2-14. 两级同步器时序



2.2.2 滤波器

如果用户希望避免出现影响系统的尖峰和毛刺，为确保消除所有毛刺，可以选择滤波器选项。使用滤波器选项时，信号会先通过两级同步器，然后再通过滤波器。

XNOR 充当多数表决功能，只要 XNOR 的输入彼此不同，输出便为 0。

- 如果 XNOR 的两个输入相等，则其输出为 1
- 如果 XNOR 输出为 1，则最后一个 D 触发器上的门控输入为高电平
- 如果 XNOR 输出为 0，则最后一个 D 触发器上的门控输入为低电平

使能滤波器后，输出将最多延迟 4 个 CLK 周期。使用这些选项时，LUT 中短于两个同步时钟周期的任何输出都将被滤除。

根据用作 LUT 输入的逻辑值，有效输出信号有时可能会在多个时钟周期内保持高电平。如果在此类情况下选择了滤波器选项，则会滤除有效信号，进而破坏系统功能。因此，在使用滤波器前，必须确保信号被滤波器延迟或缩短后不会产生负面影响。在实现任何滤波器选项之前，建议分析一下当前配置下 LUT 中的最短有效信号。如果最短有效信号短于两个周期，则不能使用滤波器。

图 2-15. 滤波器

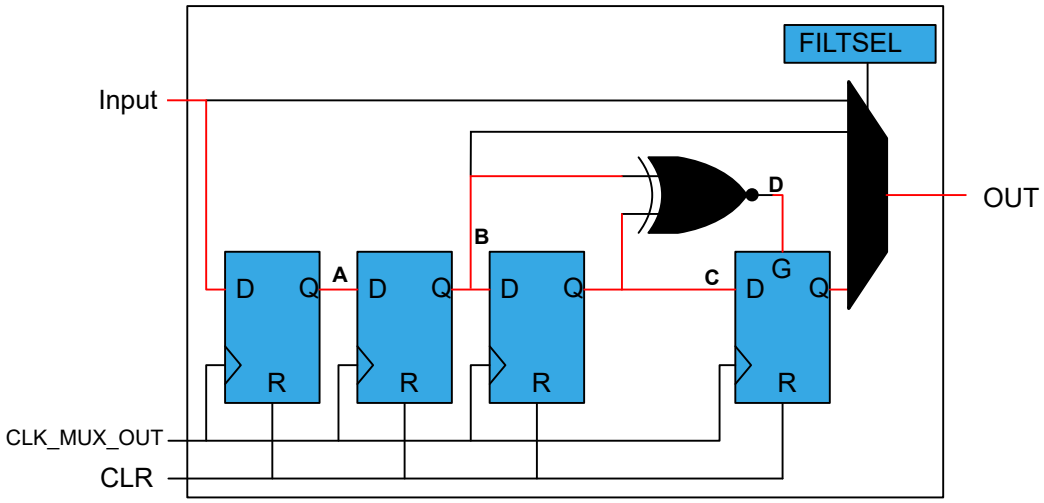
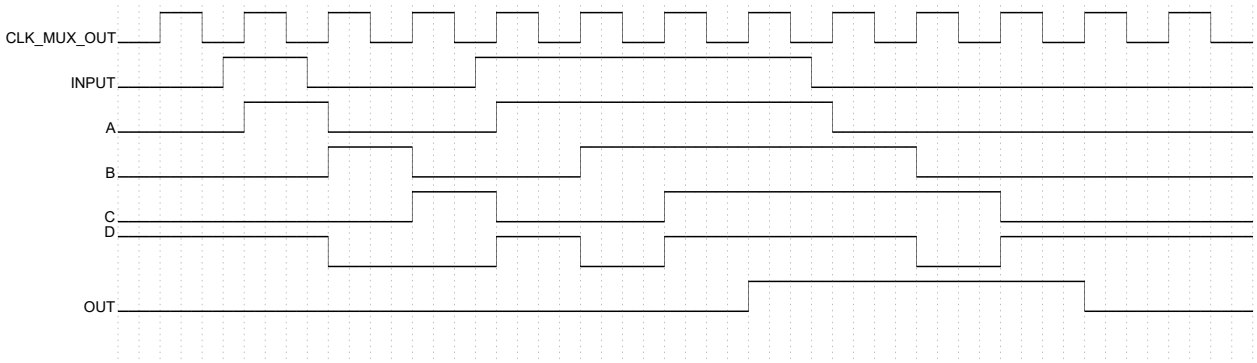


图 2-16. 滤波器时序



2.2.3 边沿检测器

如果使能边沿检测器，则可在输入端检测到上升沿时生成脉冲。要检测下降沿，可通过编程真值表提供相反电平来实现。例如，每次真值表输出高电平（1）时，事件系统便会发送脉冲以触发另一个外设（如定时器）。

图 2-17. 边沿检测器

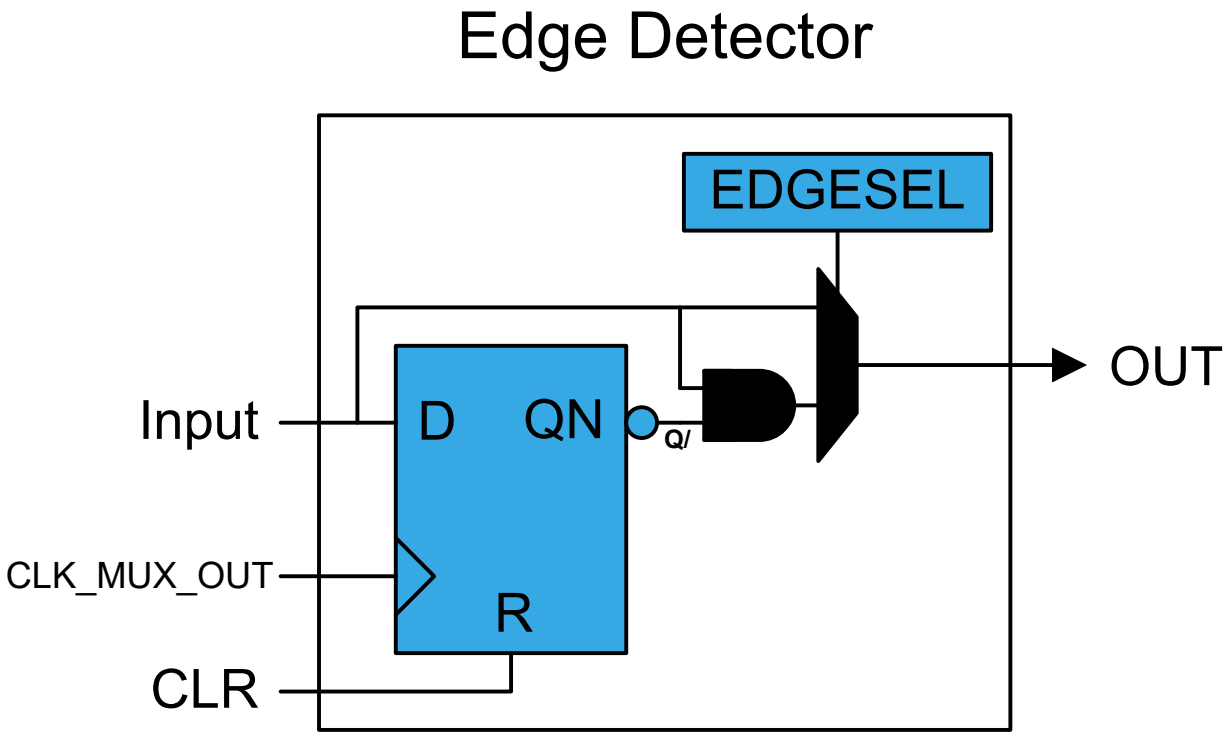
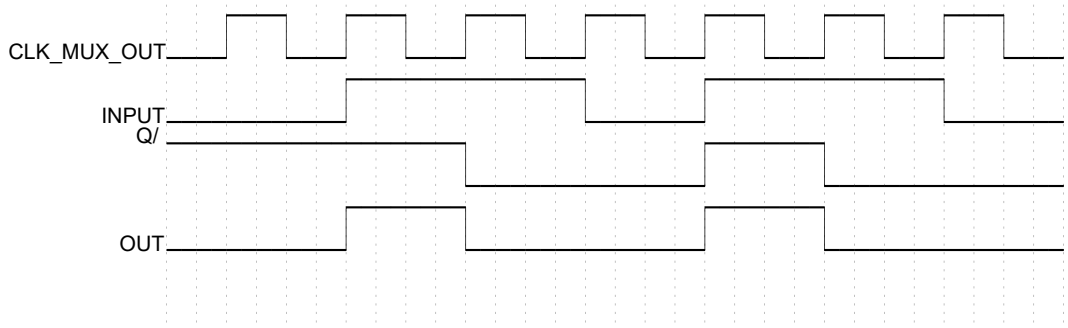


图 2-18. 边沿检测器时序



2.3 顺序逻辑

每对 LUT 都可以连接到内部顺序逻辑。顺序控制寄存器中的顺序选择位 SEQSEL 用于选择不同的顺序逻辑模块。顺序逻辑可用于在 CCL 中实现更为复杂的功能。

CCL 具有以下顺序逻辑模块：

- 门控 D 触发器（DFF）
- JK 触发器（JK）
- 门控 D 锁存器（DLATCH）
- RS 锁存器（RS）

此外，还可以使用 JK 触发器来创建 T 触发器。

在 Atmel Start 中，可以找到使用 CCL 和顺序逻辑的应用笔记和代码示例。例如：

- AVR42779 Ultrasonic Distance Measurement

2.3.1 门控 D 触发器

D 触发器（DFF）的使用非常广泛，通常称为“数据”或“延时”触发器。当 G 输入为高电平时，触发器将捕捉 D 输入的值，而该值即为 Q 输出。如果 G 输入为低电平，则将忽略 D 输入，而 Q 输出将保持前一个状态。DFF 可以看作是存储单元、零阶保持或延时线。

D 输入由偶数 LUT 输出（LUT0）驱动，G 输入由奇数 LUT 输出（LUT1）驱动。

图 2-19. 门控 D 触发器

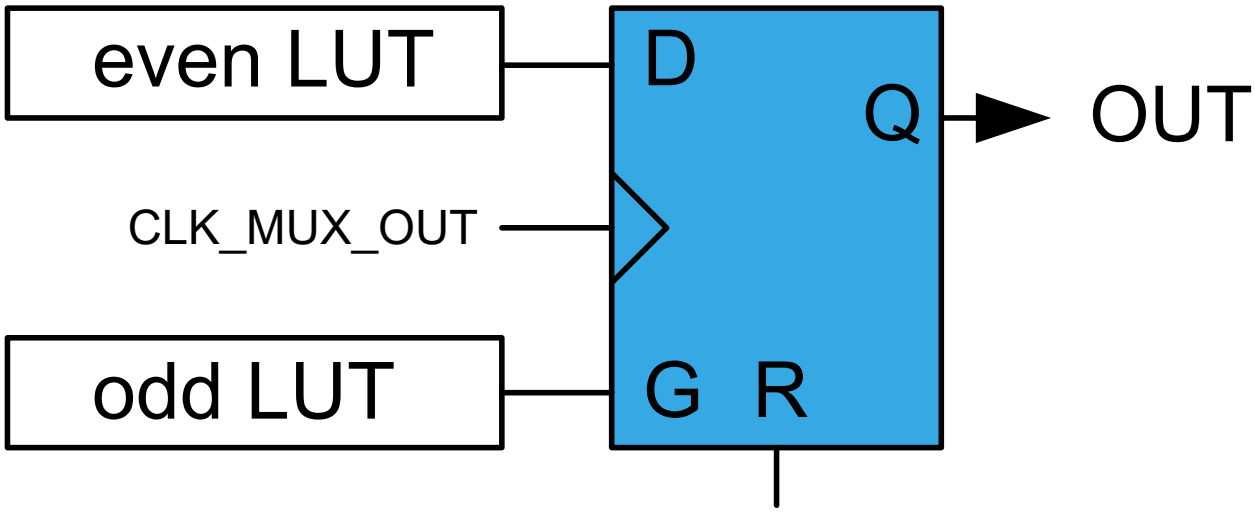


表 2-12. DFF 行为

R	G	D	OUT
1	X	X	清零
0	1	1	置 1
		0	清零
	0	X	保持状态（无变化）

2.3.2 JK 触发器

JK 触发器是所有触发器中使用最广泛的触发器，它可以配置为 SR 触发器、D 触发器或 T 触发器，因此可以视为通用触发器。当 J 和 K 相等或均为逻辑 1 时，它本质上是一个没有非法输出状态的门控 SR 触发器。

J 输入由偶数 LUT 输出（LUT0）驱动，K 输入由奇数 LUT 输出（LUT1）驱动。

图 2-20. JK 触发器

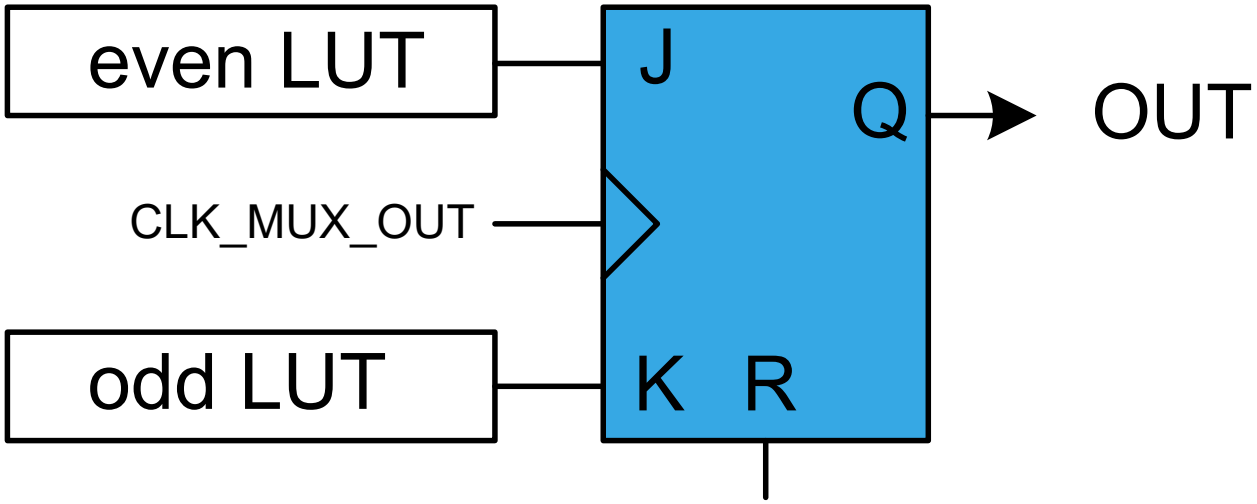


表 2-13. JK 行为

R	J	K	OUT
1	X	X	清零
0	0	0	保持状态（无变化）
0	0	1	清零
0	1	0	置 1
0	1	1	翻转

2.3.3 门控 D 锁存器

D 锁存器是一个多谐振荡器锁存器电路，当两个输入均为高电平时，没有 SR 锁存器中的非法输入状态。D 锁存器也称为透明锁存器。这意味着只要门控信号 G 为高电平，D 上的信号就会直接通过锁存器到达输出。

D 输入由偶数 LUT 输出（LUT0）驱动，G 输入由奇数 LUT 输出（LUT1）驱动。

图 2-21. 门控 D 锁存器

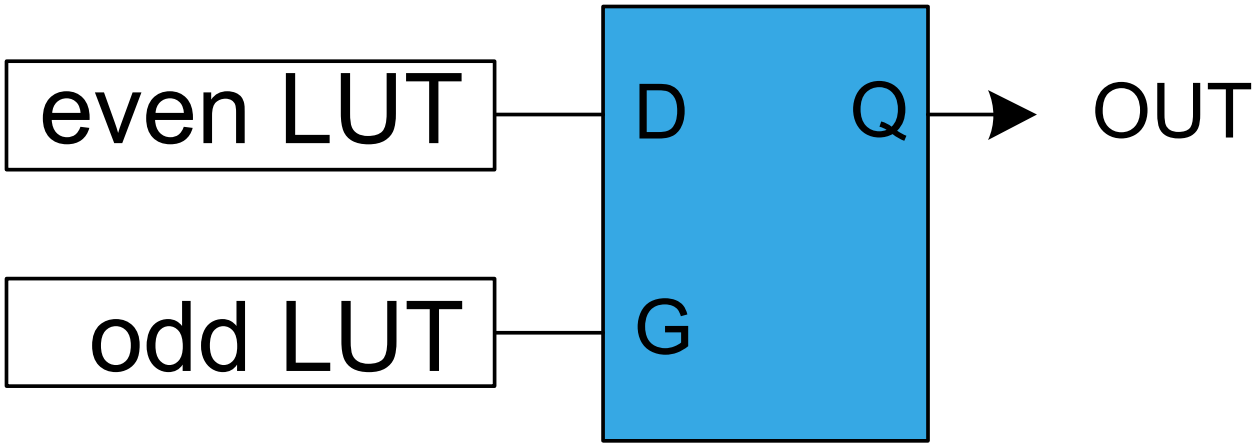


表 2-14. D 锁存器行为

G	D	OUT
0	X	保持状态（无变化）

.....（续）

G	D	OUT
1	0	清零
1	1	置 1

2.3.4 RS 锁存器

RS 锁存器的功能基本上与 SR 锁存器相同，惟独在禁止状态（S 和 R 均等于 1 时）下有所不同。在这种状态下，SR 锁存器输出将变为 1，而 RS 锁存器输出为 0。

S 输入由偶数 LUT 输出（LUT0）驱动，R 输入由奇数 LUT 输出（LUT1）驱动。

图 2-22. RS 锁存器

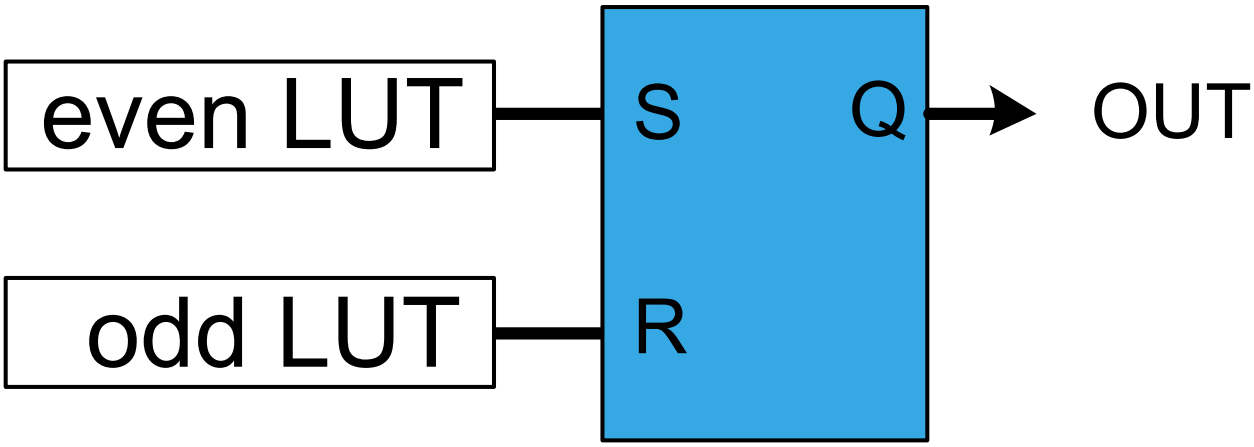


表 2-15. RS 锁存器行为

S	R	OUT
0	0	保持状态（无变化）
0	1	清零
1	0	置 1
1	1	禁止

2.3.5 T 触发器

T 触发器也叫翻转触发器，可通过将 JK 触发器上的两个输入连接到相同源来创建。这种触发器可以用作分频器。当 J 和 K 输入均为高电平时，T 触发器将在每个时钟周期翻转输出，因此输出频率将为输入频率的一半。

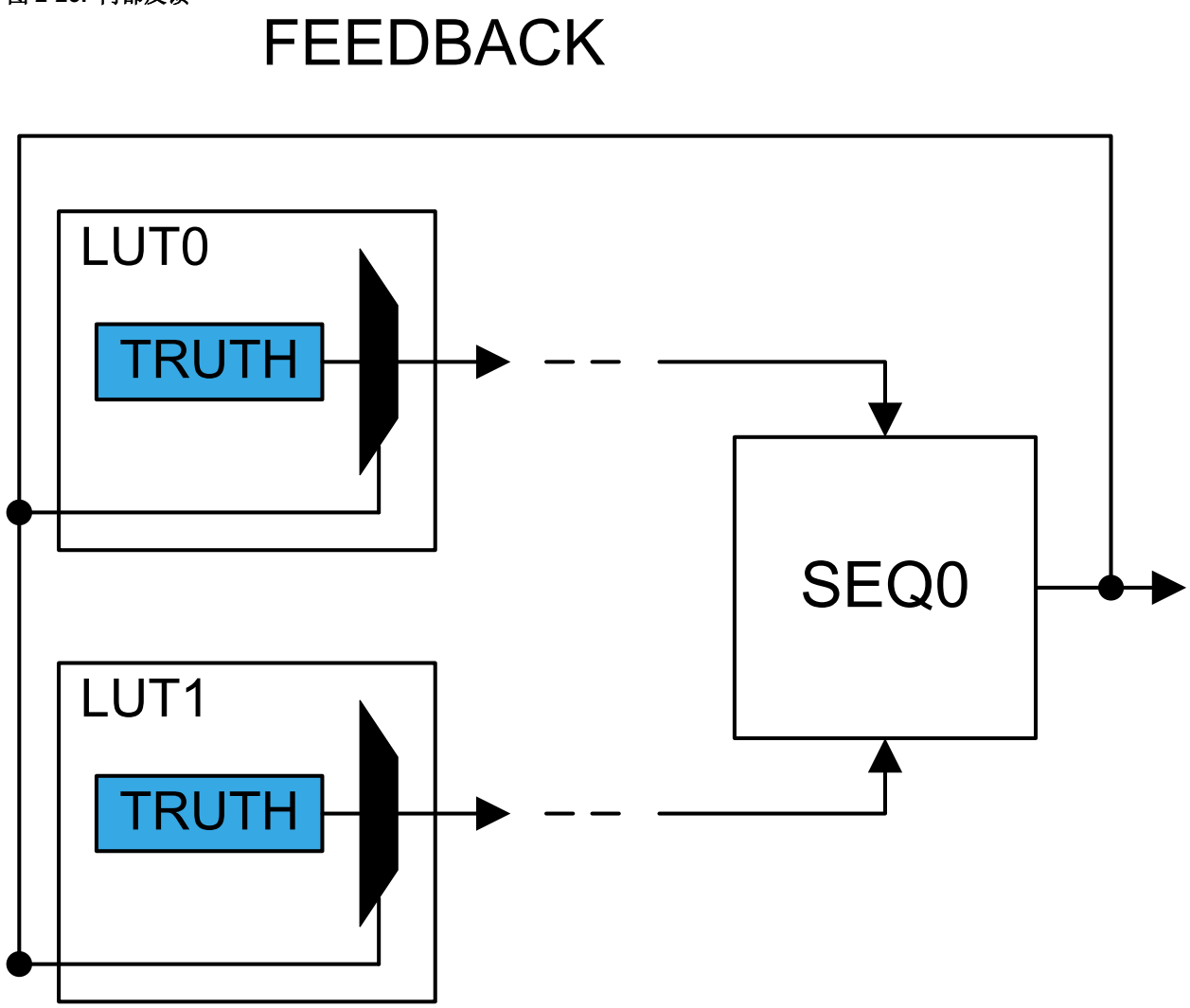
对于可能导致 JK 意外翻转的任何尖峰，可使用滤波器和边缘检测器来滤除。

2.3.6 反馈

通过将顺序逻辑的输出反馈到 LUT 的输入，可以创建一种新型器件：有限状态机（Finite State Machine，FSM）。

在某些系统中，可能需要使用反馈来实现所需的功能。如果已知系统的输出状态，将会十分有帮助，因此可以将顺序逻辑输出的内部反馈提供给两个 LUT 上的任何输入，从而使反馈系统变得非常灵活。

图 2-23. 内部反馈



3. 事件系统简介

事件系统（EVSYS）是一个典型的 CIP，当一个外设（事件生成器）的内部状态发生变化时，可通过事件系统的事件通道来触发其他外设（事件用户）中的操作。该系统简单而强大，能够自主控制外设，而无需使用中断、CPU 或 DMA 资源。它可在外设之间提供短暂且可预测的响应时间，从而降低软件复杂度、缩短软件代码长度和执行时间以及降低功耗。

AVR 通常支持多个并行事件通道，每个事件通道可以分为三个不同的部分：

- 事件生成器，具有一个或多个事件源
- 事件路由网络
- 事件用户

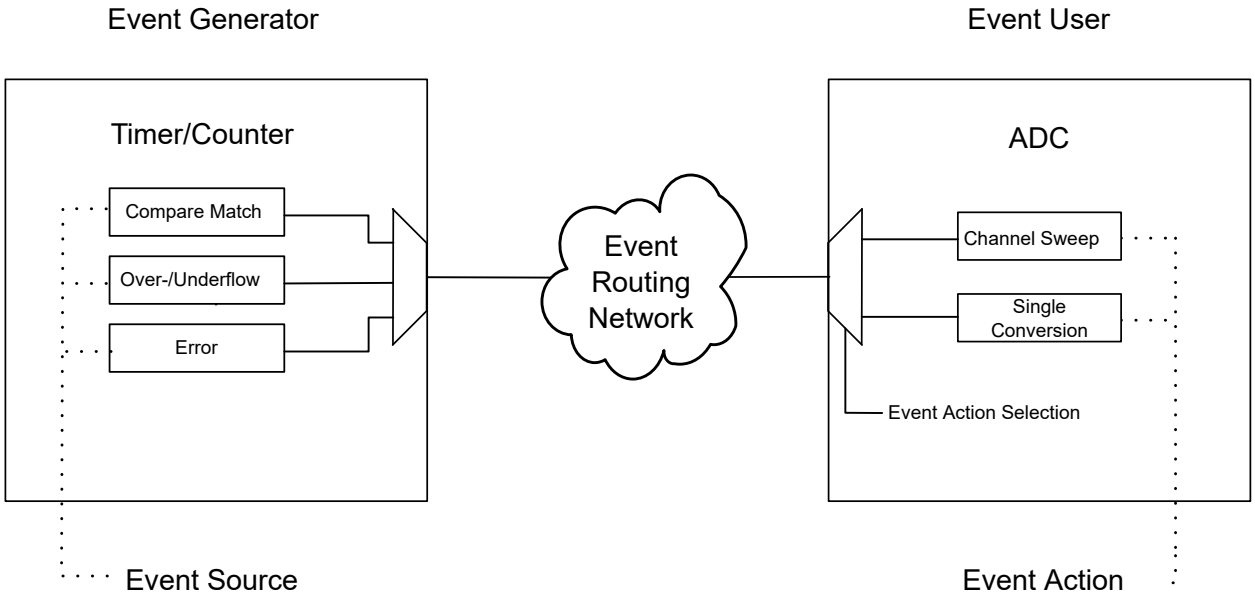
事件用于指示外设的内部状态发生了变化。能够生成事件的外设称为事件生成器。每个事件生成器能够根据外设中的多种变化生成事件。这些变化都是独立的事件源。通道既可与主时钟异步，也可与主时钟同步，具体取决于应用要求。tinyAVR® 1 系列具有四个异步事件通道和两个同步事件通道。寄存器 ASYNCH0、ASYNCH1、ASYNCH2、ASYNCH3、SYCNCH0 和 SYNCH1 分别用于为这些通道配置事件源。每个事件通道上只能路由一个来自事件生成器外设的触发信号，但多个通道可使用同一个生成器源。多个外设可以使用来自同一通道的事件。

事件路由网络负责处理从事件生成器到事件用户的事件路由。每个事件生成器的每个事件源均连接每个事件通道的输入。事件用户是一个外设模块，可以利用事件来触发操作（称为事件操作）。事件用户通过选择事件通道来选择要响应的事件源。实际事件源由所选事件通道中的倍频器设置确定。

事件系统可直接连接模拟和数字转换器、模拟比较器、I/O 端口引脚、实时计数器、定时器/计数器以及可配置定制逻辑外设。此外，也可从软件和外设时钟生成事件。

下图给出了一个简化版事件系统，其中一个定时器/计数器作为事件生成器，一个 ADC 作为事件用户。事件通道多路开关可以从三个可用源中选择一个并通过相应的事件通道进行路由。

图 3-1. 事件源、生成器、用户和操作示例



对于 I/O 寄存器和选通，事件系统使用外设时钟。此外，它还可在没有任何时钟的休眠模式下使用。一个事件通常持续一个时钟周期。

手动事件生成：可以通过软件或使用片上调试系统来生成事件。生成的事件直接注入事件通道中。事件通道无需具有与之关联的事件源即可使用手动事件生成功能。如果事件通道具有与之关联的事件源，则手动生成的事件优先，并将覆盖外设事件。以下两个寄存器用于实现手动事件生成：STROBE 和 DATA。通过写入 STROBE 寄存器来触发事件生成。生成信号通知事件时，仅需使用 STROBE 寄存器。生成数据事件时，必须同时使用 STROBE 和 DATA，并且 STROBE 必须在 DATA 之后写入。

事件和休眠模式：事件系统可以在工作模式和待机休眠模式下运行。在所有其他休眠模式下，外设模块均无法使用事件系统进行通信。

3.1 tinyAVR® 1 系列中外设的事件功能概述

下文概述了 tinyAVR® 1 系列中外设与事件相关的功能，这些功能对于开发独立于内核的应用十分有用。有关详细信息，请参见具体器件的数据手册。

- PORT——I/O 引脚控制器
  - 通过所有 GPIO 引脚生成事件
- TCA——16 位 A 型定时器/计数器
  - 对事件信号的上升沿进行计数
  - 对事件信号的两种边沿进行计数
  - 只要事件信号为高电平，就会对预分频时钟周期进行计数
  - 对预分频时钟周期进行计数。事件信号控制计数方向
  - 可以基于计数器上溢、下溢和比较匹配来生成输出事件
- TCB——16 位 B 型定时器/计数器
  - 可以通过事件信号来控制初始化、计数和捕捉
  - 对于生成输出的模式，可以将输出分配为事件信号
- TCD——12 位 D 型定时器/计数器
  - 可以基于计数器比较匹配来生成输出事件
  - 输出事件可以延迟一定的 TCD 延时时钟周期，周期数可自行配置。TCD 延时时钟是 TCD 时钟的预分频版本。
  - 计数器操作可以通过两个单独的事件输入信号以多种不同的方式进行控制
  - 可对输入事件进行屏蔽和滤波
- USART——通用同步/异步收发器
  - 可以使用输入事件信号（而非相应的 RX 引脚）作为接收器输入
- RTC——实时计数器
  - 可以基于计数器上溢和比较匹配来生成输出事件
  - 可以定期生成输出事件（每 n 个 RTC 时钟周期生成一次），其中 n 可从一组预定义的值中选择
- CCL——可配置定制逻辑
  - 每个查找表（LUT）可以将两个单独的事件作为其对应真值表的输入
  - 每个 LUT 的输出可以分配为事件信号
- AC——模拟比较器
  - 比较器输出可以分配为事件信号
- ADC——模数转换器
  - 输入事件可以触发 ADC 转换
- UPDI——统一编程和调试接口
  - 生成可用于测量系统时钟频率的输出事件

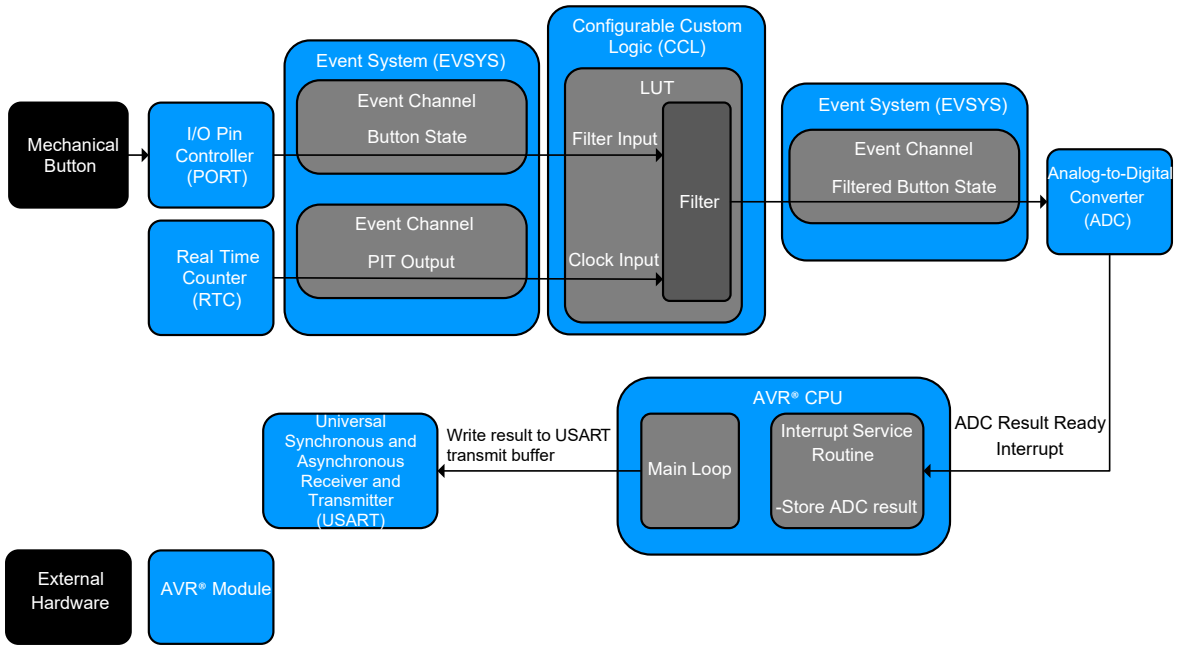
4. 应用示例——对按钮信号进行滤波并启动 ADC 转换

在许多情况下，如果将来自机械按钮的信号直接输入应用当中而不进行任何形式的滤波，会导致不可预测的行为，因为每次按下或释放按钮时，信号通常都会在高电平和低电平之间发生数次跳变。这种现象通常称为*抖动*。如果应用需要在每次按下机械按钮时都做出反应，则需要通过硬件或软件来实现某种形式的滤波（也称为*去抖*）。

本章将通过一个应用示例来说明在没有 AVR 内核干预且未添加外部滤波功能的情况下，如何实现每次按下机械按钮时都会启动单次 ADC 转换。按钮信号通过以下方式实现*去抖*：首先使用 CCL 对按钮信号进行滤波，然后使用滤波后的信号来触发 ADC 转换。信号通过事件系统进行路由，当转换结果就绪后，将通过 USART 模块传输以进行验证。

以下是所用的器件模块、CPU 以及二者间连接配置方式的示意图。欲了解有关如何在特定器件或评估工具包上实现应用程序的详细信息，请在 Atmel START 中打开并查看示例应用程序。5. 从 Atmel | START 获取源代码中介绍了如何在 Atmel START 中找到该应用程序。

图 4-1. 示例概览



4.1 事件系统（EVSYS）设置

该应用示例使用事件系统与 CCL 互相路由信号，从而实现最大的灵活性。按钮信号和合适的时钟信号必须路由到 LUT 的事件输入，而该 LUT 的输出必须路由到 ADC 事件输入。因此，在该应用中，CCL 既是事件生成器又是事件用户。

从实时计数器（RTC）中的周期性中断定时器（Periodic Interrupt Timer，PIT）单元输出的事件适合作为时钟信号，使用该事件时，器件上的其他定时器/计数器可用于实现其他用途。如果将 RTC 时钟设置为 32 kHz，则最好选择与 RTC 时钟进行 1024 分频相对应的 PIT 输出事件作为一个事件通道的事件源。这可能需要根据按钮信号的特性进行修改。随后可以将选作 LUT 的 IN[2]的输入事件配置为该通道的用户。

连接到按钮的 I/O 引脚可以配置为第二个事件通道的事件生成器。之后可以将剩余的可用 LUT 输入事件配置为该通道的用户。如果未连接外部上拉电阻，则还可以将 I/O 引脚配置为输入，并使能其相关的上拉电阻。

要通过滤波后的按钮信号来触发 ADC 转换，LUT 输出可以配置为第三个事件通道的生成器，而 ADC 可以配置为用户。

4.2 实时计数器（RTC）设置

RTC 模块包括一个称为 PIT 的功能。PIT 与 RTC 的其余部分使用相同的时钟源，使能后可提供一组时钟信号形式的输出事件，其周期为 RTC 时钟周期的 *n* 倍。可以在一组预定义事件生成器形式的事件系统中选择不同的 PIT 输出事件，每个输出事件的周期均与 RTC 时钟不同。

为了使用 PIT 输出事件，必须在 RTC 模块中使能 PIT。

4.3 可配置定制逻辑（CCL）设置

CCL 中的每个查找表（LUT）均包含一个滤波器，可用于对 LUT 输出进行同步或滤波。默认情况下，该滤波器由外设时钟信号提供时钟，但也可以使用经 IN[2]提供给 LUT 的备用时钟信号。通过在 IN[2]上提供合适的时钟信号，以及在 IN[0]或 IN[1]上提供来自机械按钮的信号，就可以使用单个 LUT 滤除按钮信号上的毛刺，如果不滤除，就会出现意外行为。

要配置 LUT 实现该用途，必须使能其滤波器和备用时钟源功能。

可以从大量不同的信号中选择 LUT 输入，其中包括两个不同的事件信号。为了最大程度地提高按钮和时钟信号源的灵活性，可以选择两个事件信号作为输入。其中一个事件输入必须分配给 IN[2]以用作备用时钟信号。另一个事件信号可以分配给其余两个输入之一，而未使用的输入可以配置为*屏蔽*。

由于在使能备用时钟功能时也会屏蔽 IN[2]，因此配置 LUT 的 TRUTH 寄存器时，只需要考虑为按钮信号选择的输入。只要按下按钮，LUT 输出就会变为高电平。例如，如果按钮信号为高电平有效并且在 IN[1]上提供，则 TRUTH 寄存器可以设置为 4。如果按钮信号为低电平有效（许多评估工具包都属于这种情况），则 TRUTH 寄存器可以设置为 1。

最后，使能 LUT 和 CCL 即可完成 CCL 设置。

如果应用需要，还可以将来自 I/O 引脚和/或其他外设的信号（非事件信号）选作 LUT 输入。

4.4 模数转换器（ADC）设置

为了使应用能够通过事件信号（而非使用内核）来启动 ADC 转换，必须使能启动事件输入（ADC 的功能）。之后，为了尽快存储和处理转换结果，还可以允许结果就绪中断。

ADC 可用的内部模拟源之一是片上温度传感器的电压。为了将 ADC 配置为对温度传感器进行采样，可以将 ADC 参考电压设置为内部参考，并且可以将传感器选作 ADC 输入信号。随后，可以将 ADC 参考电压设置为 1.1V 并在参考电压（V_{REF}）模块中使能。

按上文所述设置 ADC 后，当请求结果就绪中断时，ADC 结果寄存器中将提供转换后的 10 位电压值。要将结果转换为温度值，必须通过器件签名行中包含的失调和增益系数对其进行校正。为简单起见，本应用示例中未包含上述校正。

4.5 通用同步/异步收发器（Universal Synchronous and Asynchronous Receiver and Transmitter，USART）设置

为了便于验证和测试，可以将数据发送到串行终端进行显示。要将 USART 配置为通过 TX（发送）引脚发送数据，只需使能发送器，设置波特率并将 USART TX 引脚配置为输出即可。通过使用 Atmel START 提供的 USART 驱动程序，可以计算和配置波特率。

4.6 CPU 详细信息

由于在 ADC 中允许了结果就绪中断，并且应用示例可以通过 USART 存储和发送 ADC 结果，因此在结合将数据转发到 USART 的机制后，就可以实现正确的中断服务程序（Interrupt Service Routine，ISR）。

如果已经定义了变量 `ADC_result` 和 `send_flag`，就可以实现结果就绪中断程序，代码片段如下所示。

```
ISR(ADC0_RESRDY_vect)
{
    /*存储 ADC 结果并通知主循环发送结果*/
```

```
    ADC_result = ADC0.RESL;
    send_flag  = 1;

    /*中断标志必须手动清零*/
    ADC0.INTFLAGS = ADC_RESRDY_bm;
}
```

为简单起见，该示例仅存储和发送 ADC 结果的低 8 位。

随后，可在主循环中实现使用 Atmel START 生成的 USART 驱动程序函数来发送存储的值，代码片段如下所示。

```
/*ADC 结果已存储并准备发送*/
if (send_flag) {
    USART_0_putc(ADC_result);
    send_flag = 0;
}
```

“USART_0_putc()”函数仅将给定的 8 位写入 USART 发送寄存器。

要在器件上全局允许中断，还必须将 CPU 状态寄存器（SREG）中的 I 位置 1。

5. 从 Atmel | START 获取源代码

示例代码可通过 Atmel | START 获得，Atmel | START 是一种基于 Web 的工具，可通过图形用户界面（Graphical User Interface，GUI）配置应用程序代码。可以通过下面提供的直接示例代码链接或 Atmel | START 起始页上的 *Browse examples*（浏览示例）按钮，下载 Atmel Studio 和 IAR Embedded Workbench®对应的代码。

Atmel | START 网页： http://microchip.com/start

示例代码

- 独立于内核的外设入门：
  - http://start.atmel.com/#example/Atmel:getting_started_with_core_independent_peripherals:1.0.0::Application:Getting_Started_with_Core_Independent_Peripherals:

有关详细信息和示例项目的相关信息，请单击 Atmel | START 中的 *User guide*（用户指南）。*User guide* 按钮可以在该网页中找到，方法是在 Atmel | START 项目配置器中的仪表板视图中单击项目名称。

Atmel Studio

在 Atmel | START 的示例浏览器中单击 *Download selected example*（下载所选示例），下载 Atmel Studio 对应的代码并保存为`.atzip` 文件。要从 Atmel | START 下载文件，单击 *Export project*（导出项目），然后单击 *Download pack*（下载数据包）。

双击下载的`.atzip` 文件，将项目导入到 Atmel Studio 7.0。

IAR Embedded Workbench

有关如何在 IAR Embedded Workbench 中导入项目的信息，请打开 Atmel | START User Guide（Atmel | START 用户指南），选择 *Using Atmel Start Output in External Tools*（使用外部工具中的 Atmel Start 输出），然后选择 *IAR Embedded Workbench*。单击 Atmel | START 起始页右上角的 *Help*（帮助）或项目配置器中右上角的 *Help And Support*（帮助和支持），均可找到 Atmel | START 用户指南的链接。

6. 其他相关资源

下表列出了使用独立于内核的外设的应用笔记和 Atmel START 示例项目。

表 6-1. Atmel START 应用笔记

应用笔记	链接
Core Independent Nightlight Using Configurable Custom Logic on ATtiny1617	http://www.microchip.com/wwwappnotes/appnotes.aspx?appnote=en595063
Core Independent Brushless DC Fan Control Using Configurable Custom Logic on ATtiny817	http://www.microchip.com//wwwAppNotes/AppNotes.aspx?appnote=en592093
Digital Sound Recorder using DAC with ATtiny817	http://www.microchip.com//wwwAppNotes/AppNotes.aspx?appnote=en592092
Core Independent Ultrasonic Distance Measurement with ATtiny817	http://www.microchip.com//wwwAppNotes/AppNotes.aspx?appnote=en592094

表 6-2. Atmel START 应用示例

示例	连接
AVR®事件入门	http://start.atmel.com/#application/Atmel:Application_AVR_Examples:1.0.0::Application:Getting_STARTed_AVR_Events:
数字录音机	http://start.atmel.com/#application/Atmel:voice_recorder_with_dac:1.0.0::Application:AVR42777_Digital_Sound_Recorder:
Parrot	http://start.atmel.com/#application/Atmel:parrot_feg:1.0.0::Application:AVR42777_Parrot:
BLDC 风扇控制	http://start.atmel.com/#application/Atmel:avr42778_bldc_fan_control:1.0.0::Application:AVR42778_BLDC_Fan_Control:
超声波测距	http://start.atmel.com/#application/Atmel:cip_ultrasonic_distance:1.0.0::Application:AVR42779_Ultrasonic_Distance_Measurement:
使用 ATtiny817 事件系统	http://start.atmel.com/#application/Atmel:avr42815_using_event_system_on_attiny817:0.0.1::Application:AVR42815_-_Using_ATtiny817_Event_System:
使用独立于内核的 CCL 实现夜灯	http://start.atmel.com/#application/Atmel:core_independent_night_light_using_ccl:1.0.0::Application:Core_Independent_Night_Light_using_CCL:
使用带 TCA 和 TCB 的 CCL 进行正交解码	http://start.atmel.com/#application/Atmel:quadrature_decoding_using_ccl_with_tca_and_tcb:1.0.0::Application:Quadrature_Decoding_using_CCL_with_TCA_and_TCB:
模拟心跳声	http://start.atmel.com/#application/Atmel:cip_realistic_heartbeat:1.0.0::Application:Realistic_Heartbeat:

7. 版本历史

文档版本	日期	备注
C	2018 年 10 月	更新了“相关器件”一章中的图 1-1、图 1-2 和图 1-3。修正了语法和标点符号。
B	2018 年 2 月	更新了“相关器件”一章以添加 tinyAVR 0 系列和 megaAVR 0 系列。
A	2017 年 4 月	初始文档版本。

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。我们的网站提供以下内容：

- **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**——常见问题解答（FAQ）、技术支持请求、在线讨论组以及 Microchip 设计伙伴计划成员名单
- **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

产品变更通知服务

Microchip 的产品变更通知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，请访问 http://www.microchip.com/pcn，然后按照注册说明进行操作。

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（ESE）
- 技术支持

客户应联系其代理商、代表或 ESE 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 http://www.microchip.com/support 获得网上技术支持。

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

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc.及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc.的英文原版文档。

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 XMEGA 均为 Microchip Technology Incorporated 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、Precision Edge、ProASIC、ProASIC Plus、ProASIC Plus 徽标、Quiet-Wire、SmartFusion、SyncWorld、Temux、TimeCesium、TimeHub、TimePictra、TimeProvider、Vite、WinPath 和 ZL 均为 Microchip Technology Incorporated 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 Incorporated 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 均为 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：978-1-5224-5744-2

质量管理体系

有关 Microchip 的质量管理体系的信息，请访问 http://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. Chandler, AZ 85224-6199 电话：480-792-7200 传真：480-792-7277 技术支持： http://www.microchip.com/support 网址： http://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：45-4485-5910 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