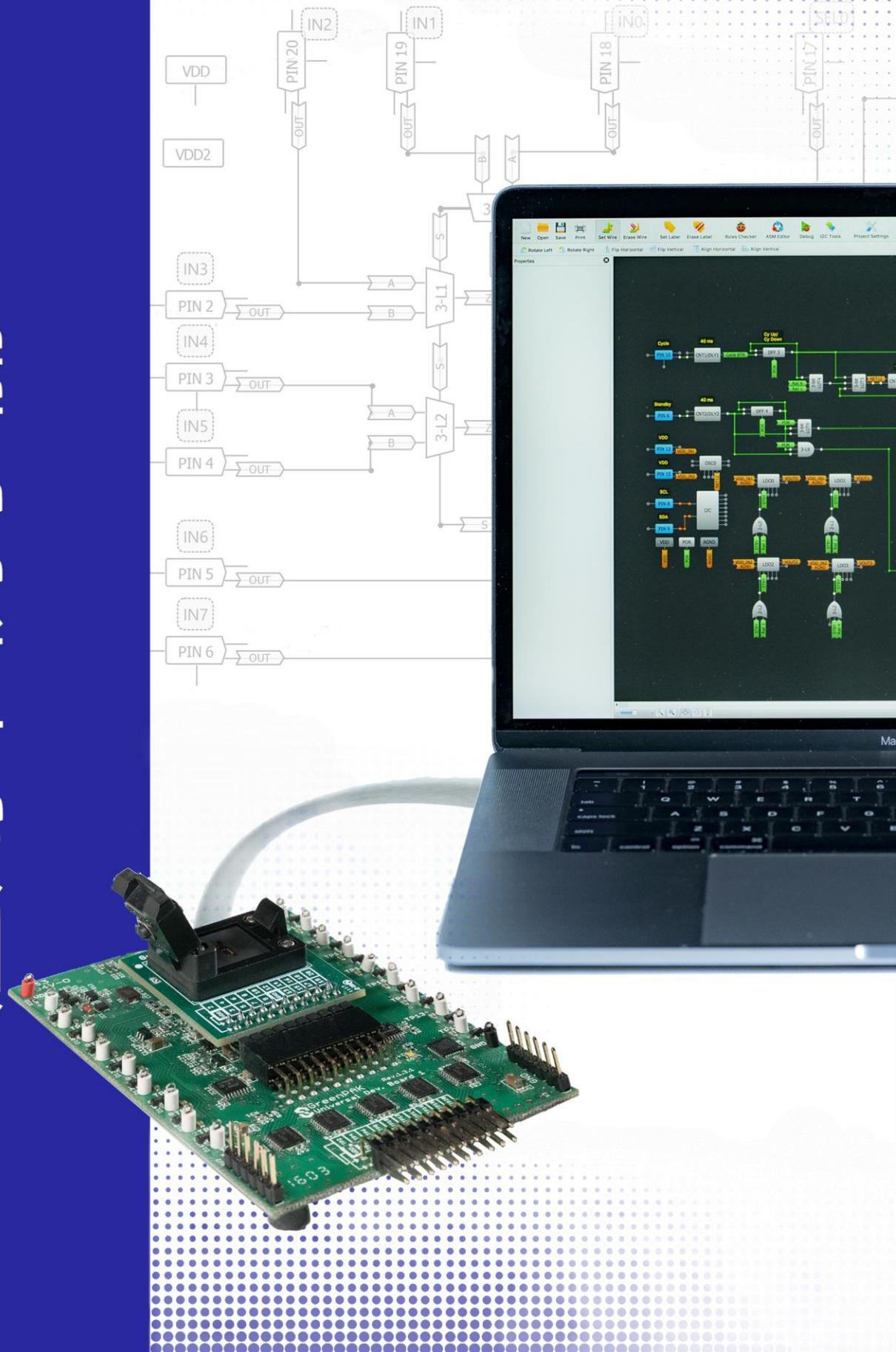
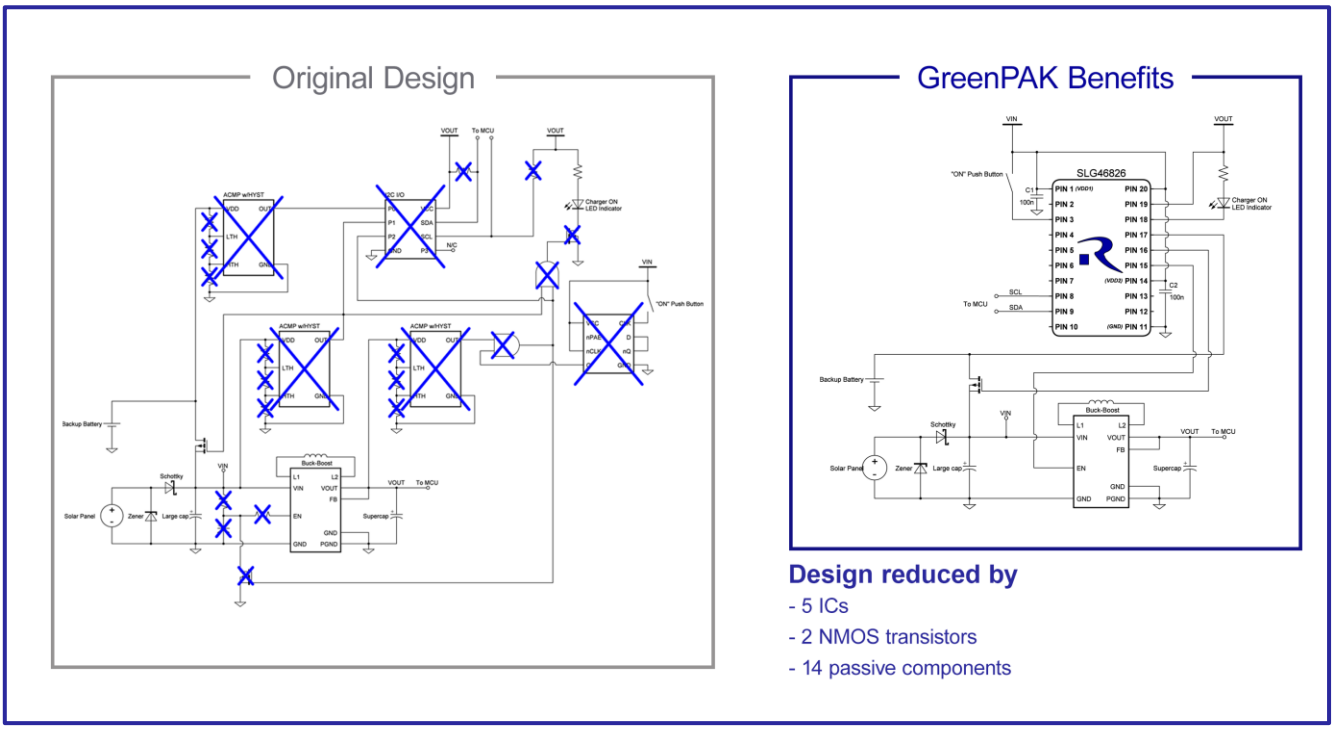
RENESAS

The GreenPAK™ 应用示例手册



GreenPAK 简介

Renesas Electronics 的 GreenPAK IC 是一系列可配置混合信号 IC [CMIC]，为系统级电路设计人员面临的常见问题提供小型、成本友好且个性化的解决方案。 GreenPAK 提供了一种显著降低 PCB 尺寸，BOM 成本和设计时间的方法。



使用 GreenPAK 降低 PCB 板尺寸的示例

由于 GreenPAK 的各种功能和可配置性，所以 GreenPAK 的应用范围非常广，凭借对产品需求正确的认知，设计人员可以将 GreenPAK 应用到大多数行业的几乎任何的应用中 。

本文档旨在加强这种认知和技术诀窍： 我们为设计人员提供应用示例手册，指出 GreenPAK 可以应用在项目中的那些地方。我们概述了不同的技巧，并提供了完整的应用示例，以帮助设计人员灵活使用 GreenPAK。

本手册的结构

本文档中每个小节都包含两种类型的内容：技巧和应用示例。 技巧专注于仅使用一个或几个宏单元来完成的任务。 应用示例部分描述了如何将这些技巧组合在一起以创建有实际价值的应用。 最简单的技巧和应用将放在第一章的开头。

每个应用对应的 GreenPAK Designer 设计文件，都可以查看、编辑。

Resources

Go Configure Software™ Hub Download

The GreenPAK Cookbook GP Files Download

Product Selection Guide

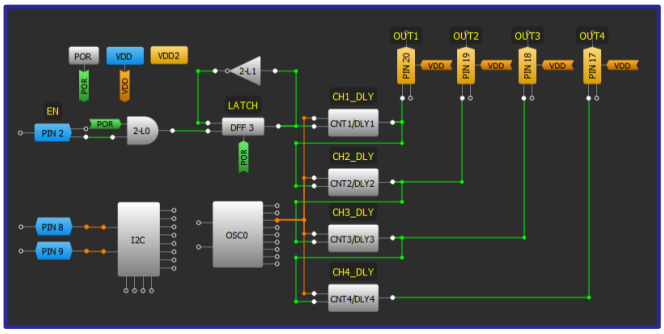
The GreenPAK Cookbook EN

The GreenPAK Cookbook JP

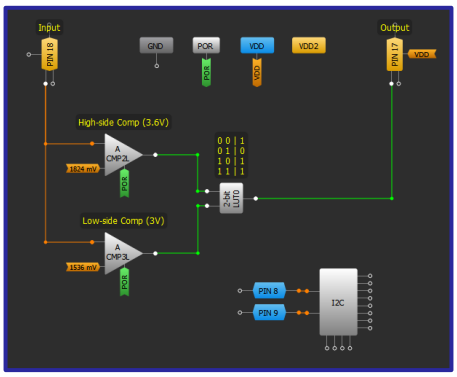
使用本手册构建你的设计

本手册列举的实例都是已经得到实际应用的工程案例。 不仅如此， GreenPAK IC 还具有本手册实例中未使用到的更多宏单元和功能。 Renesas Electronics 帮助工程师实现了数以千计的独特设计，这些设计或多或少与本手册中的案例相似或不同，但它们都是可扩展、可结合和可定制化的。

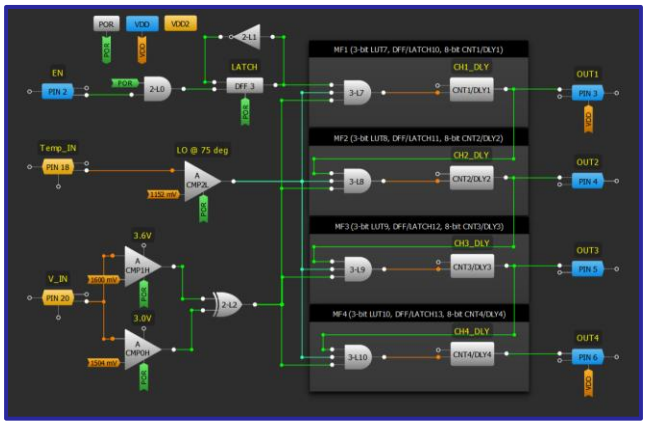
例如，电源序列发生器应用实例与第 4 章节的安全应用实例相结合，从而成为可调节的、定制化的序列发生器应用。



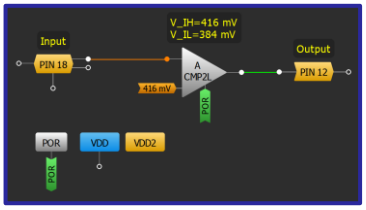
电源序列发生器



窗口比较器



独特的解决方案



过温检测

如把上述实例中的几个方案整合在同一电路中，系统将变得十分复杂且造成宏单元的浪费。而使用 GreenPAK 系列 IC 及本手册中的技巧，可以助您轻松实现设计，并且可以随意置换和修改您的设计。您可以重复使用本手册中显示的设计，或者将本文中的一些技术融入您自己的设计中。因为，这是属于你的技巧。

目录

第 1 章: 基本功能模块

本章介绍了许多在 GreenPAK 中发现的基本构建模块，它们将在整本 Cookbook 中使用。 它还将介绍一些使用查找表 look-up tables (LUTs)的简单组合逻辑设计。

技巧：如何了解宏单元的更多信息

这个技巧适用于所有版本的GreenPAK Designer。

在使用 GreenPAK Designer 时，您可能希望了解有关特定宏单元的更多信息。你可以在 GreenPAK Designer 中选中该宏单元，然后点击 **Properties** 窗口左下角的 **Information** 按钮来查看。

Info Button

概述：数字宏单元

数字宏单元是 GreenPAK 的基本功能组件，包含以下几种类型：

通用数字宏单元：

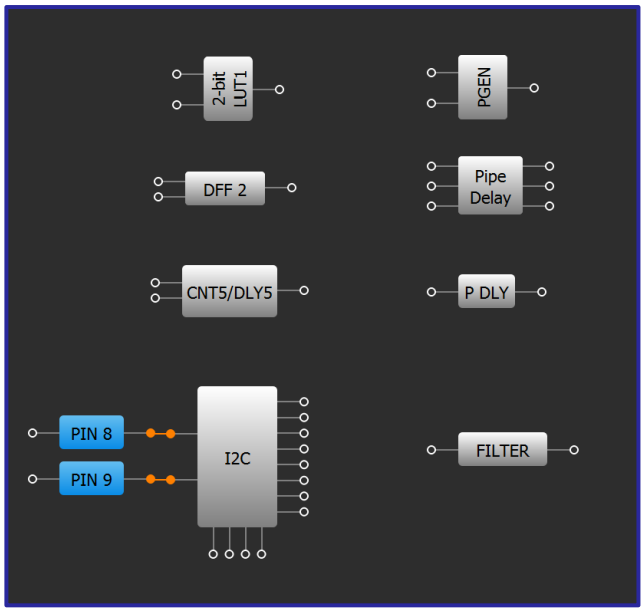
- 查找表 Look-Up Tables (**LUTs**)
- D 类触发器/锁存器 D Flip-Flop (**DFF**) / **Latch**
- 计数器/延时器 Counter / Delay (**CNT/DLY**)

通讯：

- **I²C** (大部分器件包含该单元)
- **SPI** (少数器件包含该单元)

少见：

- 模式发生器 Pattern Generator (**PGEN**)
- 管道延时 **Pipe Delay**
- 可设定延时 Programmable delay (**PDLY**)
- 滤波器/边沿检测器 **Filter / Edge Detector**



Digital Macrocells

数字宏单元

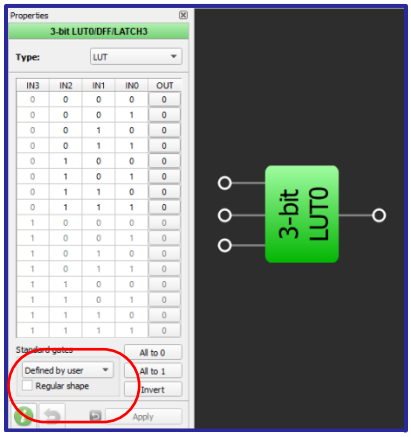
在 GreenPAK Designer 中很多的宏单元都是多用途的，你可以配置其中的一种功能作为该宏单元的功能，宏单元有哪些可供选择的功能可以从该宏单元的名字得知，比如 **2-bit LUT0/DFF/LATCH0** 暗示它可以被配置成查找表、D 类触发器或者锁存器，使用 **Properties** 窗口中的 **Type** 选项来选择宏单元的功能 。

技巧：配置标准逻辑

此技巧适用于所有 GreenPAK 芯片。

查找表[LUT]是在 GreenPAK Designer 中可配置的两输入、三输入或者四输入逻辑宏单元，可以在 **Properties** 窗口中编辑逻辑配置。

大多数的标准化逻辑已经在 GreenPAK Designs 中实现，比如 MUX、AND、OR 等，为了加快这些常见逻辑的配置，在 **Properties** 窗口中有一个 **Standard gates** 选项可以自动将查找表转换成标准逻辑门配置。如果 **Regular shape** 选项没有被选中，查找表的形状将会以标准的逻辑门符号显示。



配置 3-bit LUT0

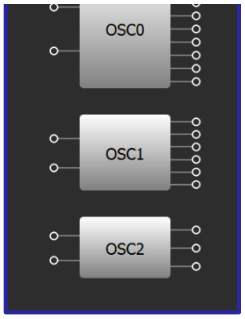
概述：振荡器

GreenPAK IC 至少包含 2 个振荡器[OSC]，很多 GreenPAK 芯片，比如 SLG46826 包含 3 个振荡器，常见的未分频频率有：

- 2KHz 低速低功耗振荡器
- 2MHz 中速振荡器
- 25MHz 高速振荡器

每一个振荡器都有几个可选择预分频系数的输出。通过配置 **Auto-power on** 可以让你在不需要时钟的时候自动关闭振荡器以降低功耗。

更多关于振荡器的信息，可以在组件被选中时，通过在 **Properties** 窗口中点击 **Information** 按钮来查看。

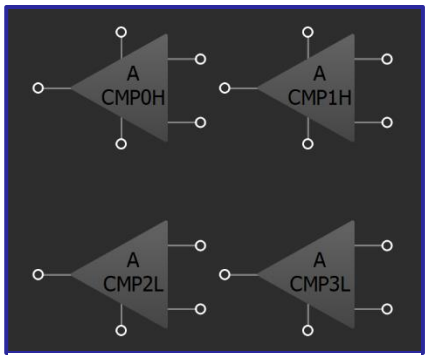


振荡器

概述：模拟比较器

大多数 GreenPAK 都包含 2 个或者更多的模拟比较器 [ACMP]，每个比较器都有一个 **IN+** 和一个 **IN-** 输入源, 可以在 **Properties** 窗口中选择输入源。

更多关于模拟比较器的信息，可以在组件被选中时，通过在 **Properties** 窗口中点击 **Information** 按钮来查看。



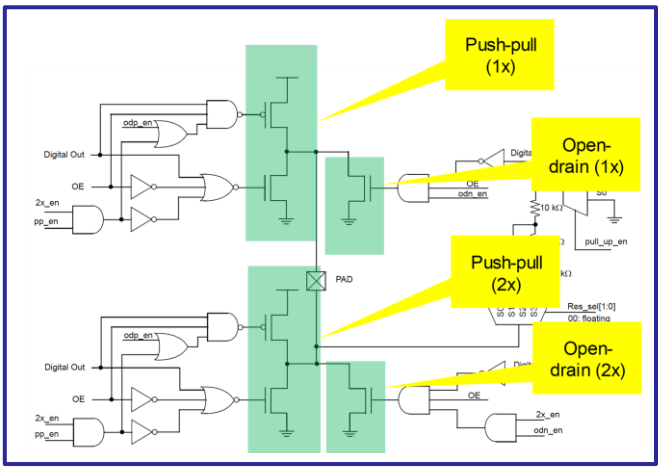
模拟比较器

概述：I/O

GreenPAK 的 I/O 是非常灵活的。无论是在同一个芯片内或是不同芯片之间，每个输入输出脚的属性均不相同，因此选型时必须先考虑输入输出脚可以实现的功能与所需要的数量。

输出可以配置成推挽输出、N 型或 P 型开漏输出，可以选择不同的输出能力， 比如 **2x**,表示两倍的输出驱动能力，还可以选择上拉电阻或者下拉电阻，有 **10kΩ**、**100kΩ** 和 **1MΩ** 可选。

输入也有多种配置可选：数字输入、施密特数字输入、低电平数字输入和模拟输入。模拟输入可以作为模拟比较器的输入源。

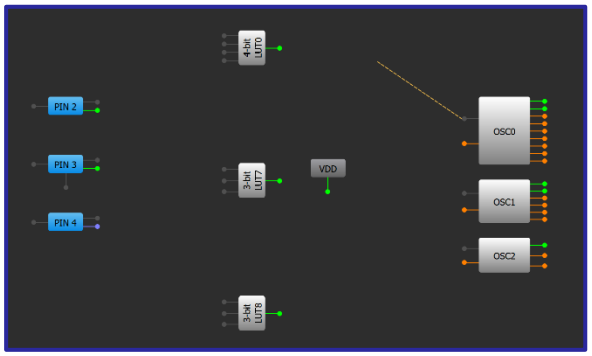


典型的 I/O 结构

概述：连接

在 GreenPAK Designer 中连接非常容易，软件会自动指示你那些连接是允许的，当你点击一个连接点时：

- 会绿色高亮所有可以连接的连接点。
- 给你一根类似橡皮筋的线，你可以拉伸到任何一个被绿色高亮的连接点。
- 如果一个连接已完成会用一根绿色的线来指示。



连接

技巧：使用 GreenPAK Designer 进行仿真

Emulation 适用于所有 GreenPAK 芯片，Simulation 适用于大部分 GreenPAK 芯片。

GreenPAK Designer 中的工具栏

当进行一项设计时，非常重要的一点是可以快速进行功能测试。

可通过以下两种方式检验你的设计：

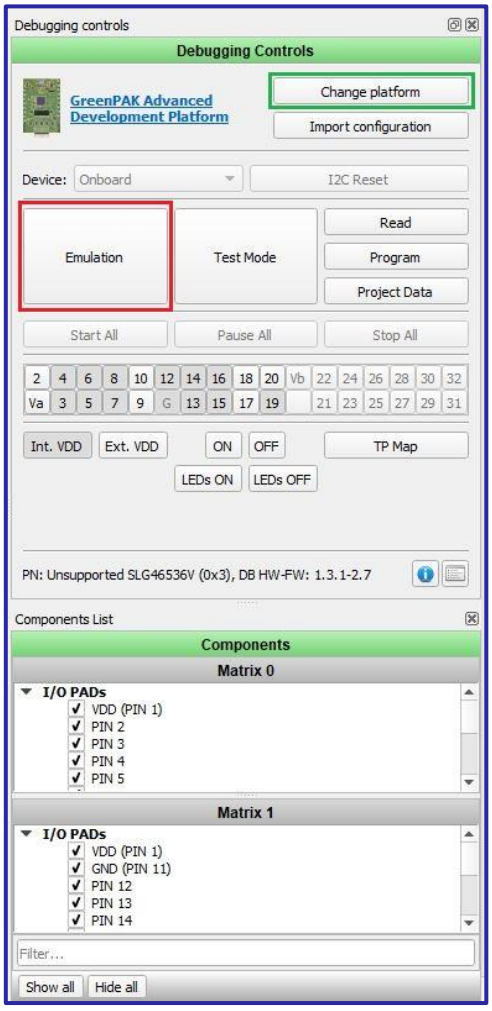
1. Simulation
2. Emulation

Simulation 在没有芯片的情况下，使用 GreenPAK Designe 模拟电路的实际性能。需要注意的是软件模拟不能完全识别真实系统的所有细节(极少数特殊条件下会存在细微差异，需要使用芯片实际测试确认)。

Emulation 使用 GreenPAK 芯片和相应的演示板，提前检验设计中的问题，避免量产中因永久性烧录造成的损失。这可以让您快速地对项目进行更改，并使用仿真来验证您的猜测。

1. 当设计完成后，单击 Debug 按钮（上图中的红色按钮），进入仿真/模拟界面。
2. 选择将要使用的验证平台。
3. 完成平台设置后, 将出现 Debug 菜单，根据您选择的平台，将提示进一步的操作。
4. 如果需要更改仿真平台，您可以随时通过单击 Change Platorm 按钮来进行更改。

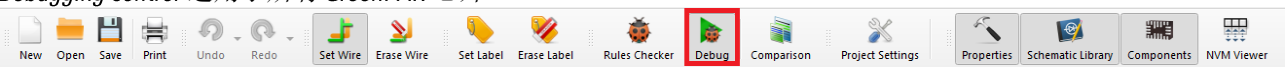
平台选择菜单



Debugging 菜单

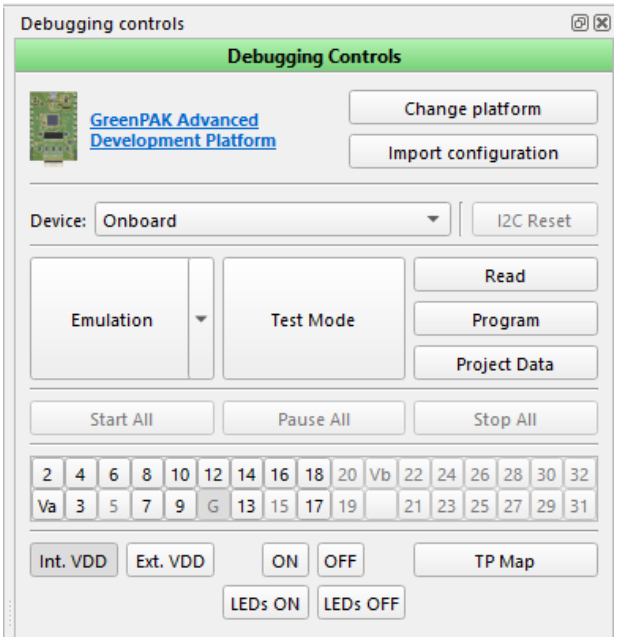
技巧：GreenPAK 烧录

Debugging control 适用于所有 GreenPAK 芯片。



GreenPAK Designer 软件的工具栏

当进行一项设计时，非常重要的一点是可以快速进行功能验证。GreenPAK Designer 软件使调试变得更加高效便捷。



Change platform
选择对应功能的硬件平台。

Import configuration
从其他硬件平台导入仿真配置。

Device
允许用户在指定的设备地址上使用外部芯片。

I²C Reset
假设与设备建立了 **I²C** 串行通信。在这种情况下，可以将设备重置为初始上电条件，包括所有宏单元的配置以及 Connection Matrix 提供的所有连接。通过将寄存器 **I²C** Reset 位设置为"1"来实现复位，复位时芯片将重新启用 Power-On Reset (POR) 序列，包括从 NVM 重新加载所有寄存器数据。

Emulation
- Emulation，当前设计将加载到芯片中(但未烧录)，并准备好在硬件板上进行测试。
- Emulation (sync), 除了 Emulation(仿真)之外，项目中所做的每个更改都将立即加载到芯片上。

Test mode
Test mode(测试模式)用于连接或断开芯片的输入/输出引脚与用户配置的 TP controls。打开 Test mode 和内部 VDD 按钮，用户可以使用测试模式来检查已烧录的芯片，而无需仿真。测试模式可以在芯片不上电的情况下工作，用户可以手动控制电源开关。

Read
通过硬件开发板读取芯片数据。

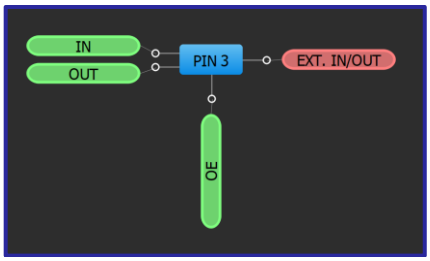
Program
将当前设计烧录到芯片中。对于某些芯片型号，用户可以通过点击 Program 按钮中的 Programming 选项来设置烧录过程。可以选择的烧录选项有：
- Program NVM (对芯片的 NVM 进行烧录)
- Program EEPROM (对芯片的 EEPROM 进行烧录)

Project Data
NVM 和 EEPROM 数据表（适用于特定芯片版本）。

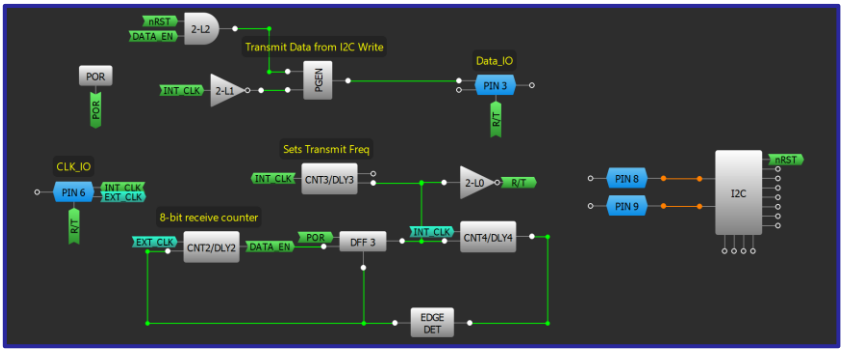
技巧：OE 脚

此技巧适用于所有带有OE 引脚的GreenPAK 芯片。

通常，GreenPAK IC 的 I/O 脚可被配置为输入或输出脚。大多数的带有**输出使能 OE** 端子的引脚的 GreenPAK IC 都可以被设置为 Digital Input 和 Digital Output 两个状态的动态转换，带有输出使能 OE 端子的引脚 GPIO, 当 GPIO 配置为输入时，OE 脚接地(或逻辑低电平)；当 GPIO 配置为输出时，OE 脚接 VDD (或逻辑高电平)。可以通过内部逻辑单元矩阵连接到 OE 脚，控制 GPIO 在 Digital Input 和 Output 两个状态间动态切换。

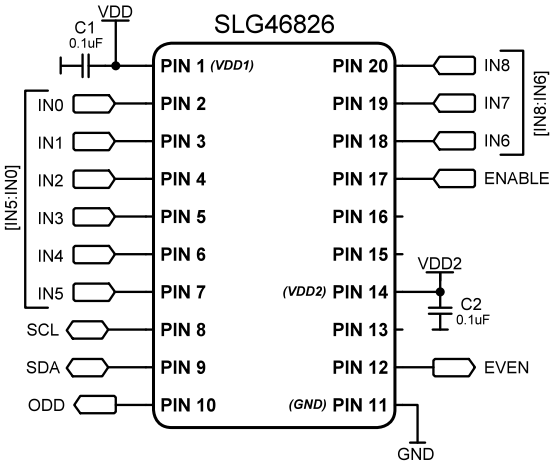


GPIO 脚可被配置为 **Digital Input/Output** 双向工作模式。此外，GPIO 脚还可设置为高阻态。
如果 GPIO 被配置为 Digital Input/Digital Output 双向工作模式，需要为 OE 提供一个模式切换的时序电路。在如下所示的电路中，CLK_IO 和 Data_IO 的 OE 脚在 CNT2 收到 8 个时钟周期后，由低电平切换为高电平，即 GPIO 被配置为输出脚进而输出内部信号。再经过 8 个时钟周期后，OE 转为低电平，重新设置 GPIO 为输入脚从而接收外部信号。



应用：奇偶校验位发生器

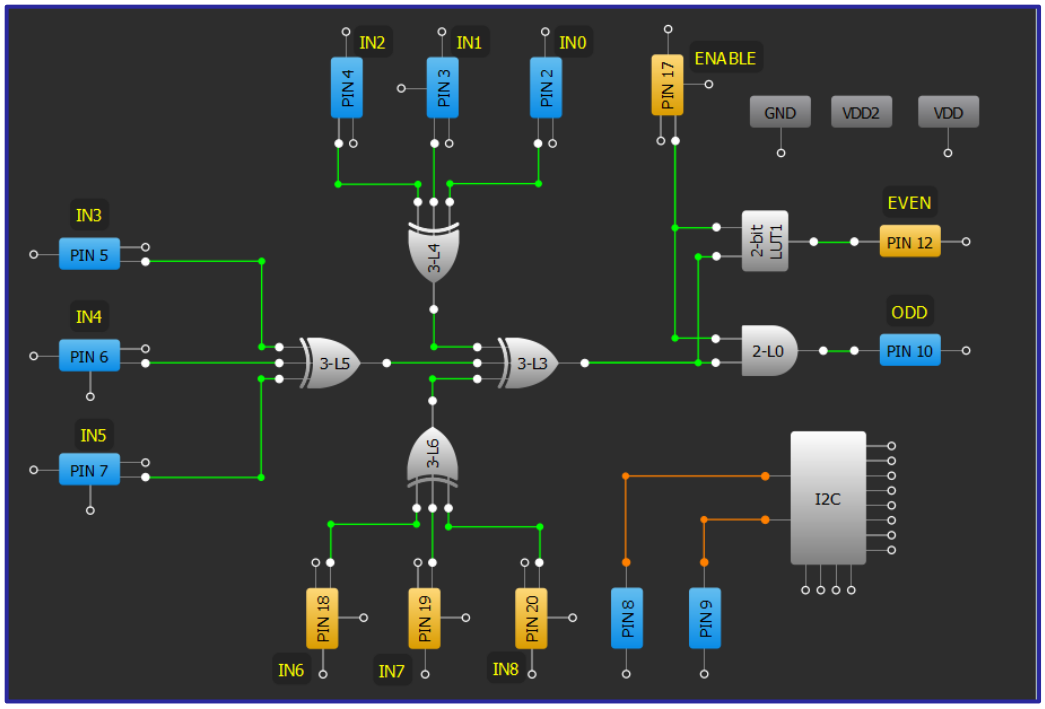
奇偶校验位生成器可以用来校验信息的完整性，是循环冗余校验最简单的实现，奇偶校验位在将数据提交给 MCU 或其他控制单元之前使用，以确保输入数据未被干扰破坏。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

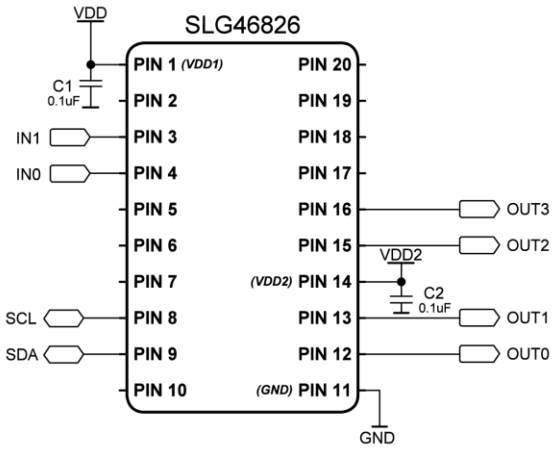


设计步骤

1. 使用技巧：配置标准逻辑 将所有输入通过 XOR 门来连接，XOR 门被用来计算 1 的个数。
2. 添加 **ENABLE** 信号逻辑。

应用：独热编码器（一位有效编码器）

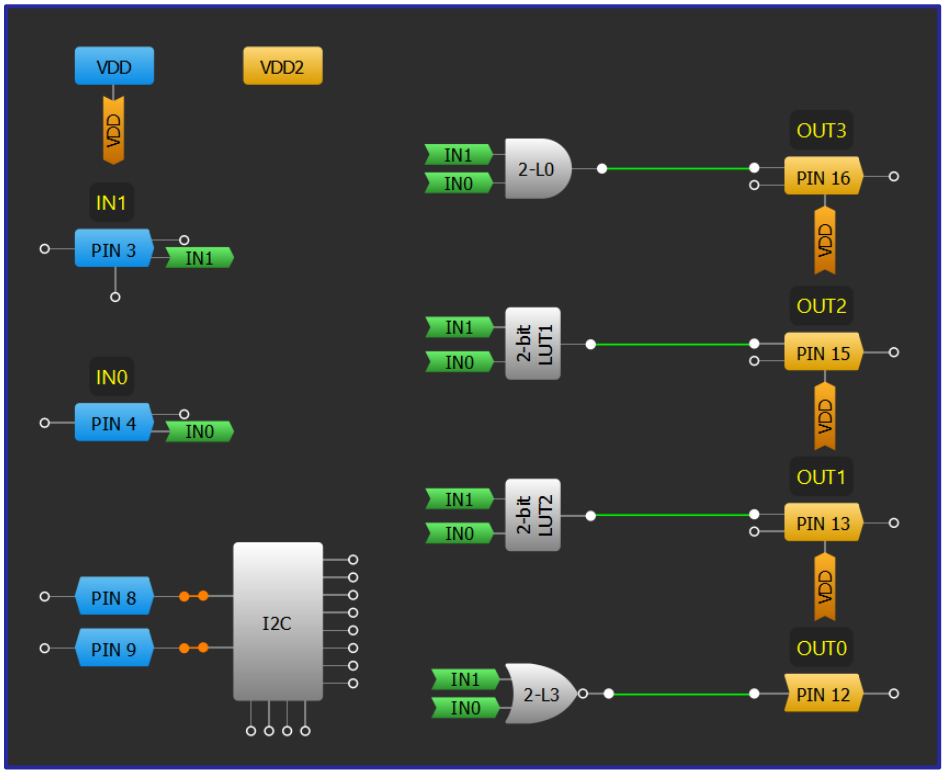
One-Hot 编码，一组数位中仅有一位为高电平(1)其他数位皆为低电平(0)，例如0001。 与之相反，一组数位中仅有一位为低电平（0），其他数位皆为高电平（1），称为One-cold 编码。本设计是一个基于输入的两位编码从而输出一个特定的独热编码的编码器。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

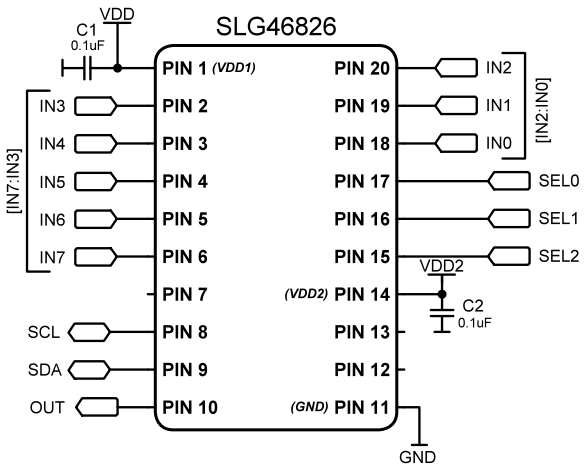


设计步骤

1. 将 2 个输入脚配置为 Digital Input(**INx**), 4 个输出脚配置为 Digital Output(**OUTx**) 。
2. 将 4 个 2-bit LUT/DFF/LATCH 配置为每个 IN 信号组合只有其中一个 **LUT** 输出高， 例如当 **IN0** 和 **IN1** 都为 0 时，只有 **2-L3** 输出才为 1，其他 LUT 均输出 0。

应用：8 位多路复用器

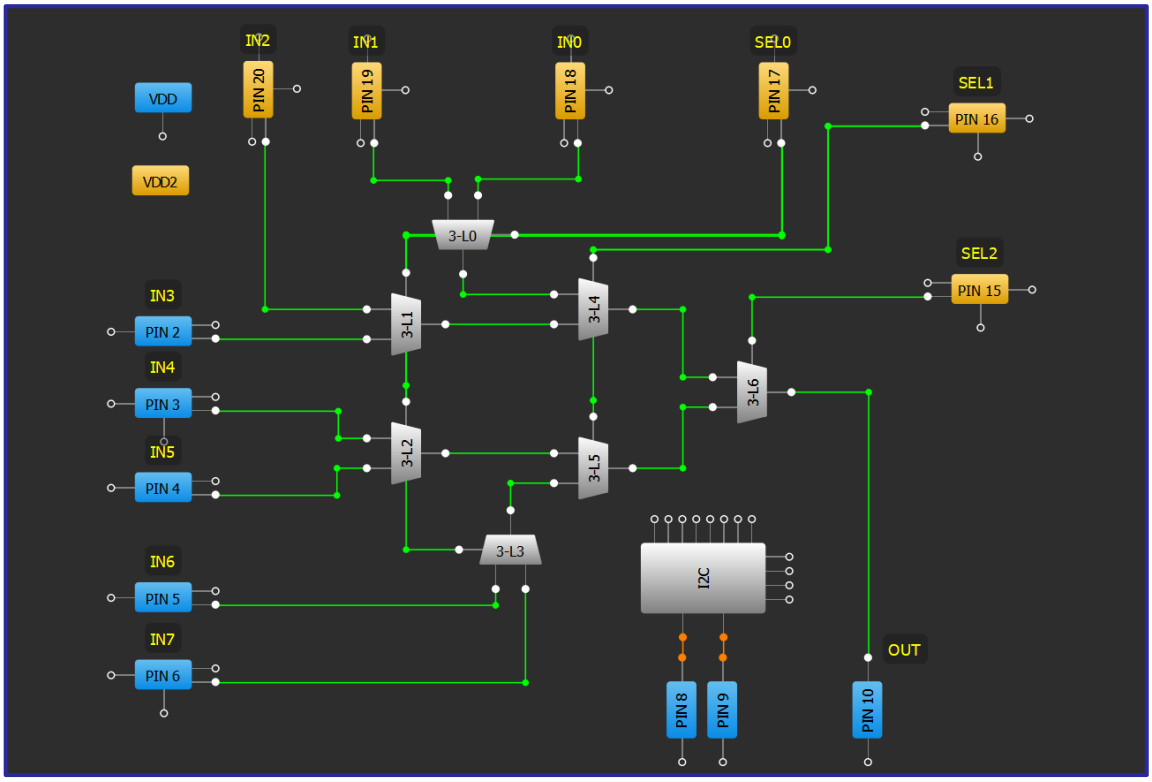
多路复用器[MUX]用于从多路输入信号中选择一个作为输出，用 GreenPAK 实现的MUX 传输延时可以和分立式逻辑 IC 相当，为纳秒级别。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

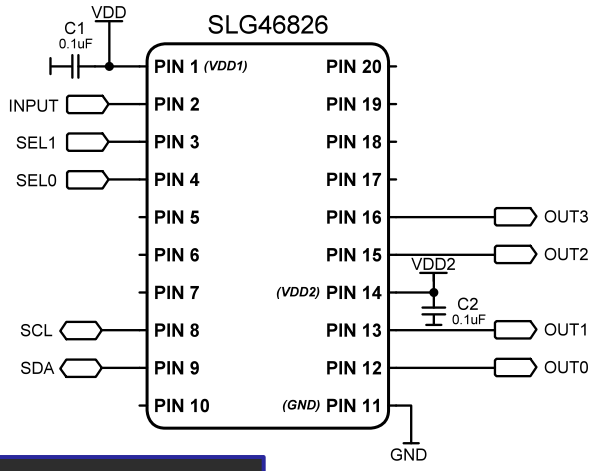


设计步骤

1. 使用 技巧：配置标准逻辑 将 4 个 3-bit LUT0/DFF/LATCH 配置为 **Multiplexer**，然后将 **Multiplexer** 的 **A 输入**或 **B 输入**与各路输入相连接, 最后将 4 个 **Multiplexer** 的 **S** 都连接到 **SEL0** 位。
2. 添加第二级和第三级 Multiplexer 以创建更高位的 **SEL** 位[SEL1 和 SEL2] 。
3. 将最后一级 Multiplexer 的输出信号连接至输出脚。

应用：多路输出选择器

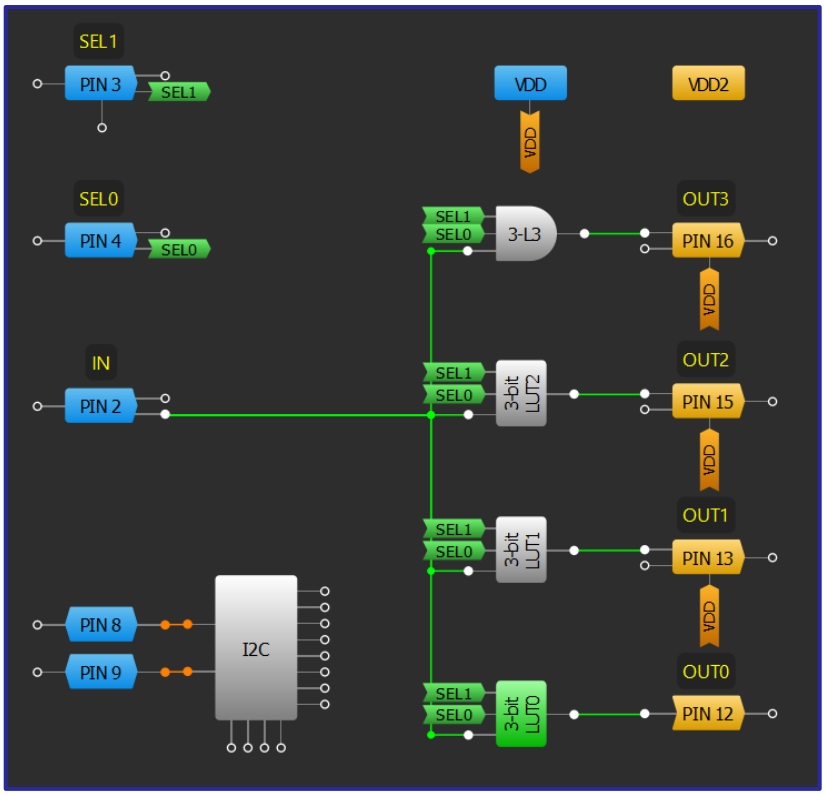
多路输出选择器用于从几个通道中选择其中一个(或几个)作为一个输入信号的输出通道。 在需要通过单根线发送几种不同类型的数据应用中，多路输出选择器被使用。这在通信系统中很常见。



所需元器件

- 任意 GreenPAK IC
- 无需其他元器件

GeenPAK 设计图



设计步骤

1. 配置输入脚为 1 个信号输入（**IN**）,2 个选择线（**SELx**）及 4 个输出脚（**OUTx**）配置输入引脚。
2. 将 LUT 配置为每个通过选择线上的特定逻辑输入传递来自 **IN** 的信号。 例如，当 **SEL0**、**SEL1** 和 **IN** 为 HIGH 时，**3-L3 输出**为 HIGH。

第 2 章:时序逻辑

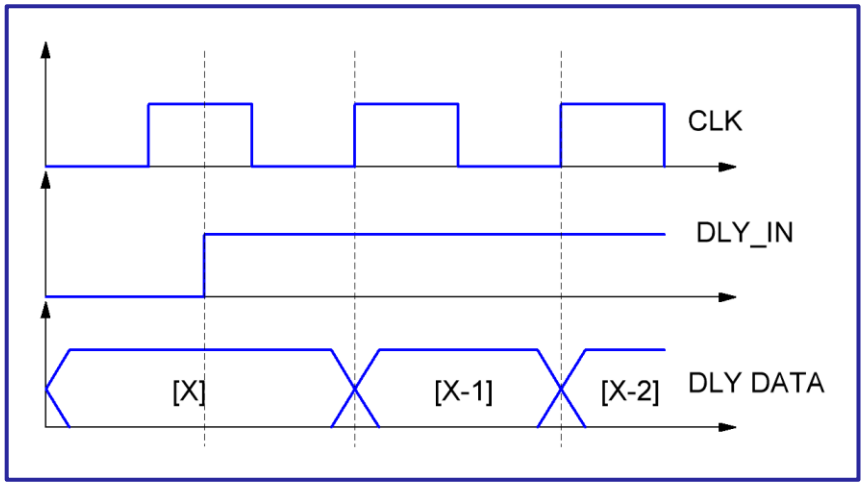
本章会介绍涉及时序逻辑的应用。 一些时序逻辑应用程序是指计数器、系统复位电路、电源时序器和状态机。

技巧：优化 CNT/DLY 的精度

此技巧适用于所有GreenPAK，振荡器和CNT/DLY 的精度因器件而异。

和所有带内部振荡器的芯片一样，GreenPAK 在时序上也带有固有的偏差，这归因于制造、温度以及用户设计等因素。通过遵循一些简单的设计规则，可以提高 GreenPAK 中 Counter 和 Delay 的精度。

应当考虑振荡器和 CNT/DLY 之间的联系，振荡器是全局的，可以用在所有的 CNT/DLY 中，而且不会和 counter 和 delay 的起始和结束信号同步，因此 counter 或者 delay 在使能信号到来之后，只有等到振荡器的下一个时钟沿到来之后才会真的开始，就如下图所示，一个 delay 的使能信号[DLY_IN]的上升沿发生在时钟电平[CLK]的中间，所以只会等到下一个时钟上升沿到来之后[DLY DATA]才会开始向下计数减一。



上升沿延时时序

将这个因素考虑进去之后，CNT/DLY 延时时间计算公式如下：

$$Delay_{time}(typical) = \frac{(Counter\_Data + 1) + t}{clock}, where\ t\ is\ between\ 0\ and\ 1$$

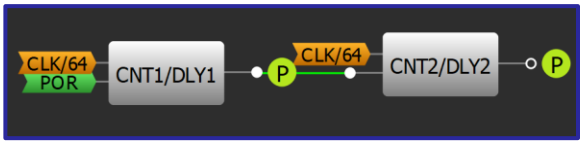
因此随着 Counter data 值的增大，t 对延时时间的影响将会减小，使用了更大的 Counter data 之后，如果要保持延时时间不变，就需要使用一个更快的时钟 [更大的F Clock]，在 **Properties** 窗口，可以为 CNT/DLY 配置 **Counter data** 和**时钟源**。

此外，参考各个 GreenPAK 数据表内关于时序的特性时，应当考虑 Power-ON 时间、频率稳定时间、温度变化等因素所造成的偏差。

技巧：CNT/DLY 的级联

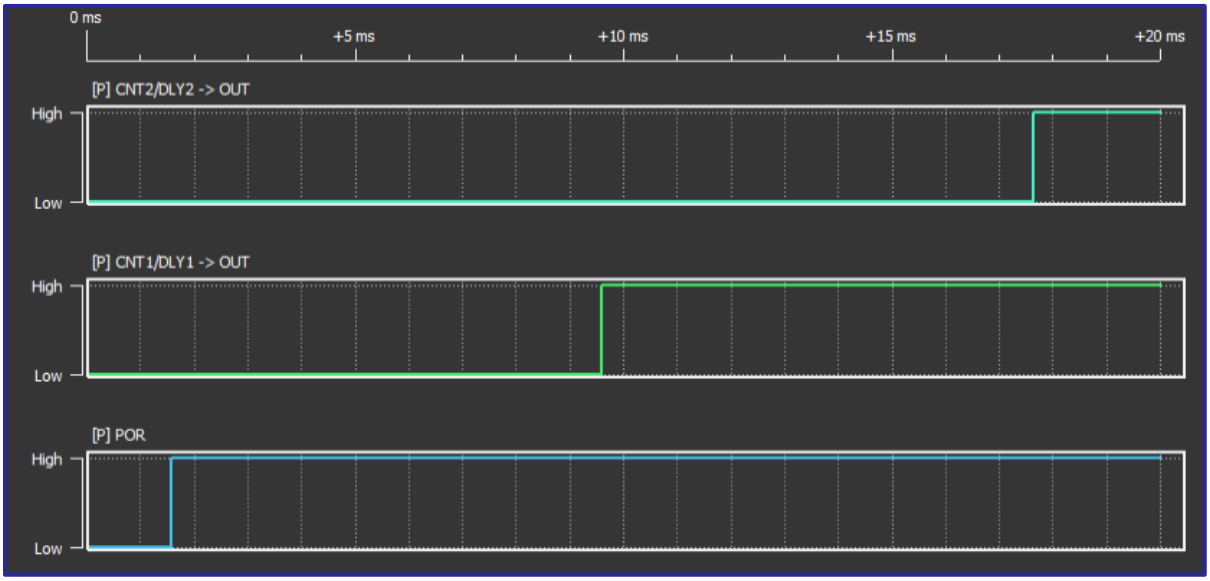
此技巧适用于所有 GreenPAK。

DLY 可以进行级联，通过把前一个 DLY 的 OUT 连接到后一个 DLY 的 DLY_IN 的方式可以将一连串的 DLY 连接起来，也就是 DLY 的级联。



级联上升沿延时

先在 **Properties** 窗口的 **Mode** 选项里选择 Delay 模式，通常还需要将所有 DLY 的 **Edge select** 设置为一样， 下图显示了将 2 个设置为上升沿延时 8ms 的 CNT/DLY 级联作用于 **Power-ON-Reset (POR)** 信号上的效果。



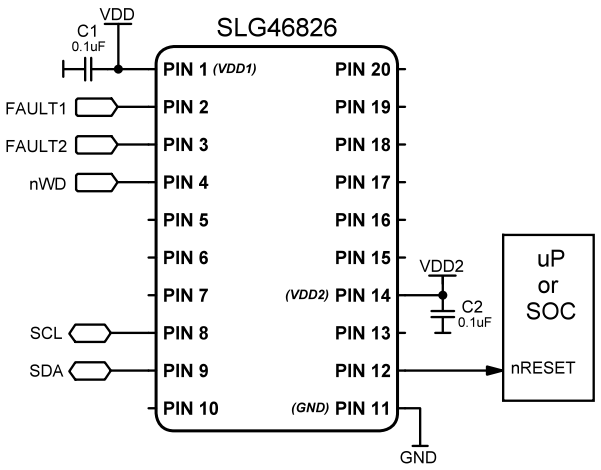
级联延时仿真

CNT 同样可以级联以获得更长的计时，CNT 是通过将前一个 CNT 的 OUT 连接到后一个 CNT 的 CLK 上来实现级联的，先用鼠标选择后一个 CNT，然后在 **Properties** 窗口的 Connections 栏的 **Clock** 选项里选择前一个 CNTx/DLYx 的 OUT 作为时钟源。

级联计数器

应用：系统复位

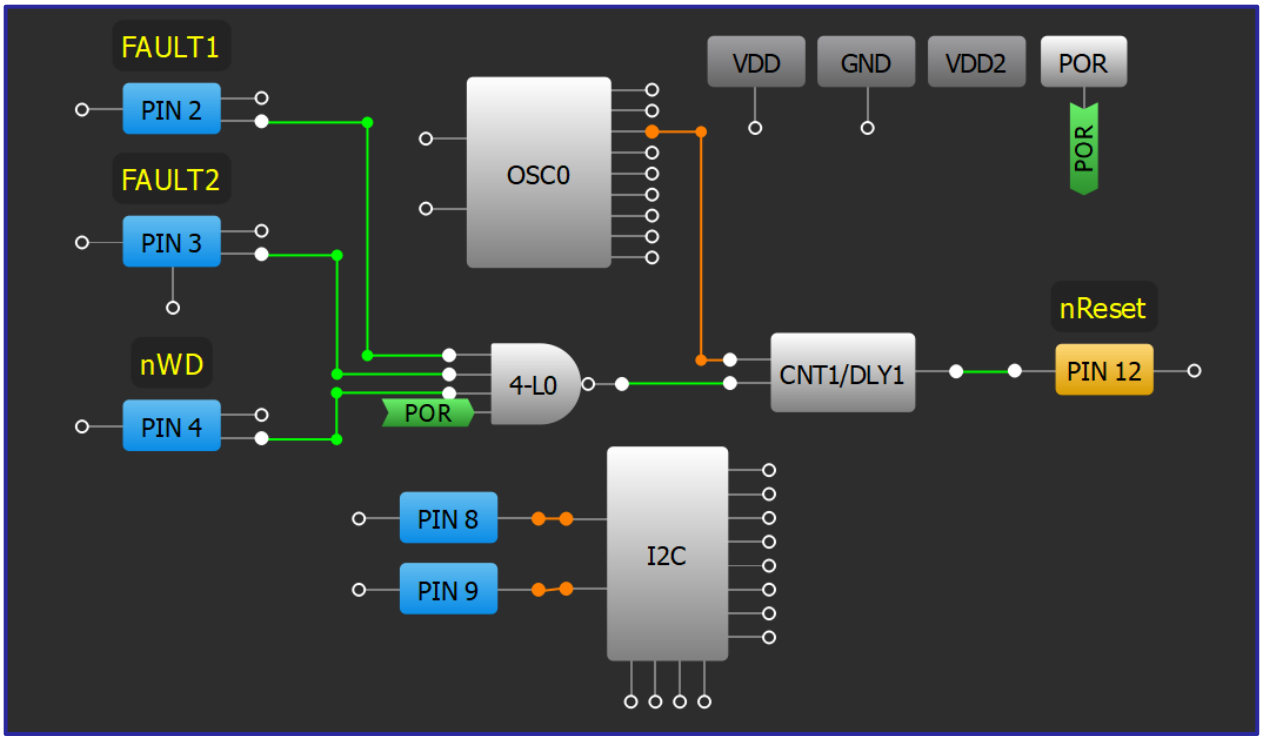
复位 IC 被用在故障、手动复位、欠压等情况下为微处理器提供一个复位信号。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

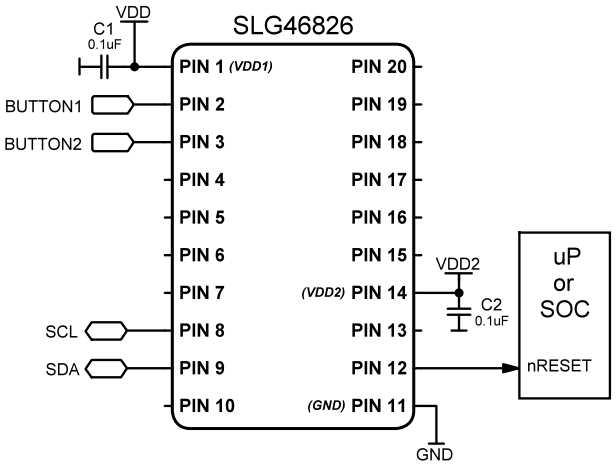


设计步骤

1. 将输入信号的 I/O 配置为 **Digital input**。
2. 添加 LUT 逻辑单元以在任何复位信号激活时发出正确的复位信号，逻辑关系取决于高电平有效还是低电平有效。
3. 配置一个 **CNT/DLY** 模块为 **One shot** 模式，**Edge select** 为 **Rising**，根据所需的脉冲宽度设置相应的 **Counter data**，对于低电平有效的脉冲，将 **Output polarity** 选择为 **Inverted(nOUT)**。
4. 将 **CNT/DLY** 模块的 **OUT** 连接至输出引脚。

应用：多按钮复位

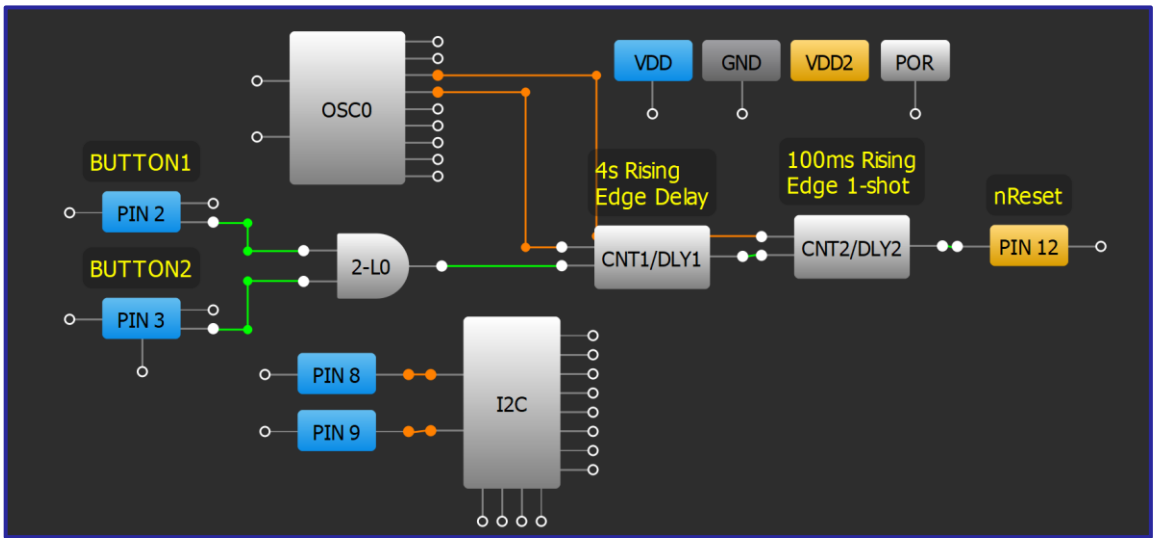
同时按住多个按钮来触发复位是许多设备常见的交互方式，用一个独立的IC 实现此功能是为了确保系统中的其他部分遇到一个或者多个软件、固件或者硬件故障时，复位仍然能够得到确认和响应。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

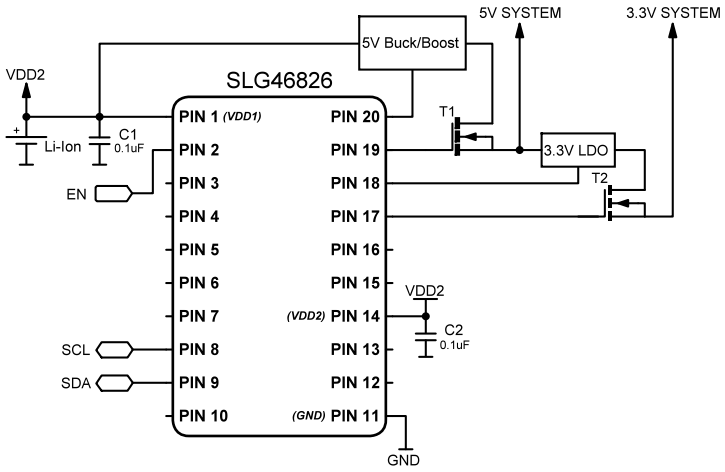


设计步骤

1. 将每个按钮对应的 I/O 配置为 **Digital input**。
2. 添加 LUT 逻辑单元以在所有按钮都处于活动状态时发出正确的复位信号，逻辑关系取决于高电平有效还是低电平有效。
3. 将一个 **CNT/DLY** 模块配置为 **Delay** 模式，**Edge select** 选为 **Rising**，根据按钮所要求按下的时间长度设置相应的 **Counter data**，对于低电平有效的脉冲，将 **Output polarity** 选择为 **Non-inverted (OUT)**。
4. 将一个 **CNT/DLY** 模块配置为 **One shot** 模式，**Edge select** 选为 **Rising**，根据复位信号所需的脉冲宽度设置相应的 **Counter data**，对于低电平有效的脉冲， **Output polarity** 选择 Inverted(**nOUT**)
5. 将 **CNT / DLY** 模块的 **OUT** 连接至输出引脚。

应用：电源序列发生器

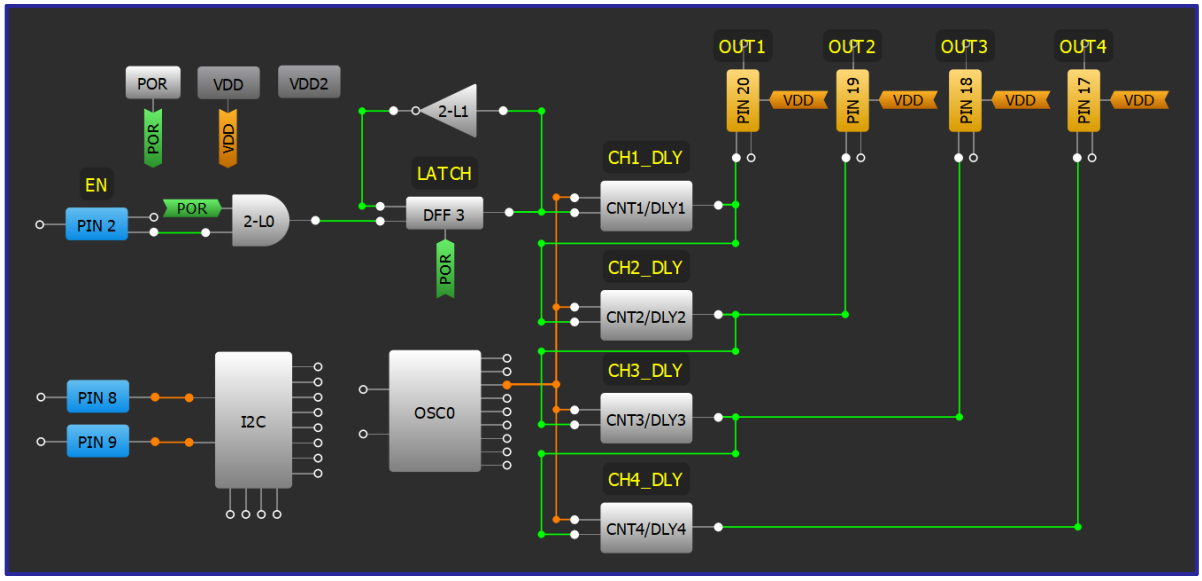
当设计人员需要按顺序激活系统的不同部分时会用到电源序列发生器。这对于有多个电源轨的系统来说至关重要。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

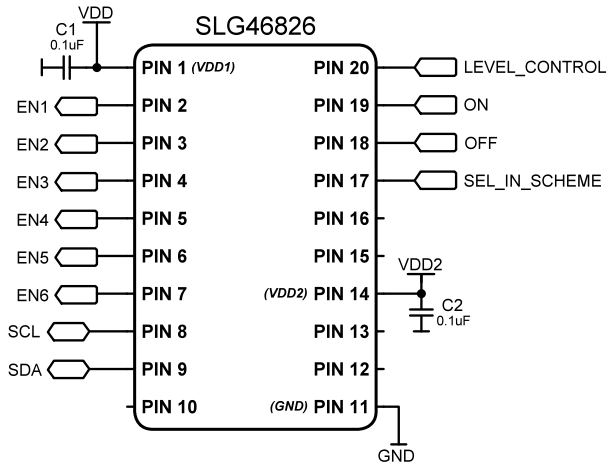


设计步骤

1. 按照所需的启动条件配置 LUT。
2. 使用一个锁存器或 DFF 来维持启动信号，以便 DLY 模块可以读取它。
3. 使用 技巧：CNT/DLY 的级联 将各个 DLY 模块级联。
4. 将各个 DLY 模块的 OUT 连接到相应的输出引脚。

应用：电源时序控制器

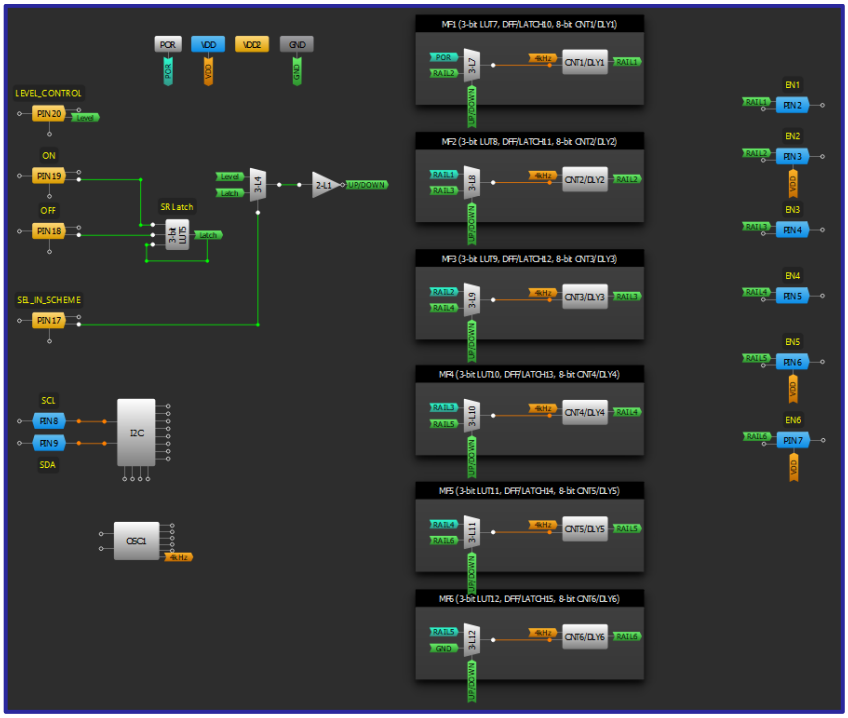
当设计人员需要按顺序激活系统的不同部分时会用到电源序列发生器。这是一种很典型的电源时序，只有当其他电源轨关闭后，这个控制芯片的电源才会关闭。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

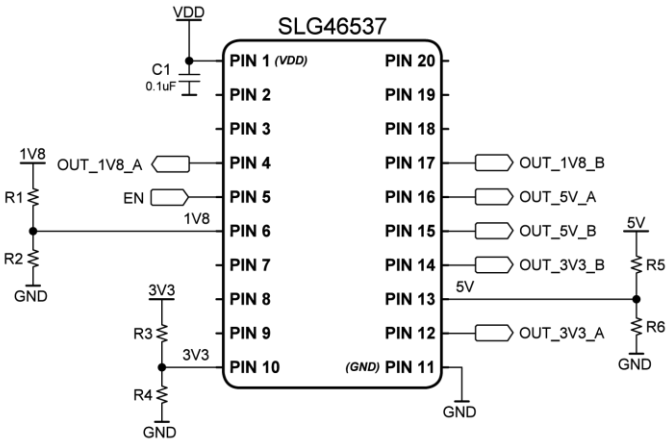


设计步骤

1. 配置输入信号源，可以通过 **PIN17** 在 Level 和 Lactching 两个模式间进行选择。
2. 通过多路选择器 Multiplex 将控制信号输入给到 DLY 模块，使用这样一个多功能复用模块达到级联时序的效果。
3. 将各个 DLY 模块的 OUT 连接到相应的输出引脚。

应用：电压监测电源时序控制器

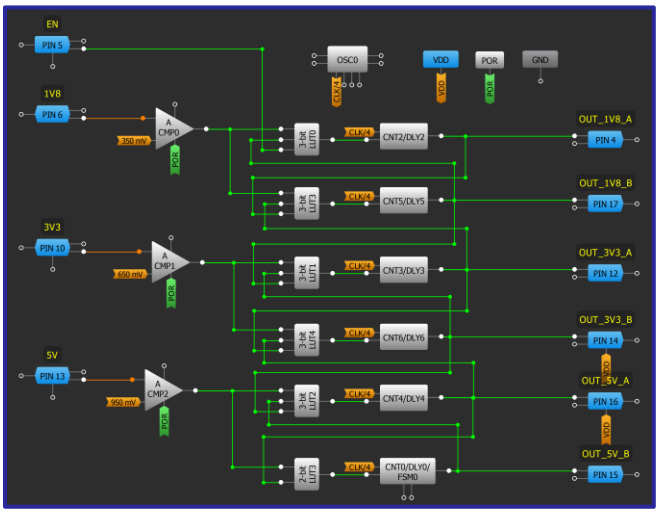
设计人员可以通过序列时序来实现按顺序激活系统的各个部分， 通过监控电源轨和故障条件来调整系统所需的时序。

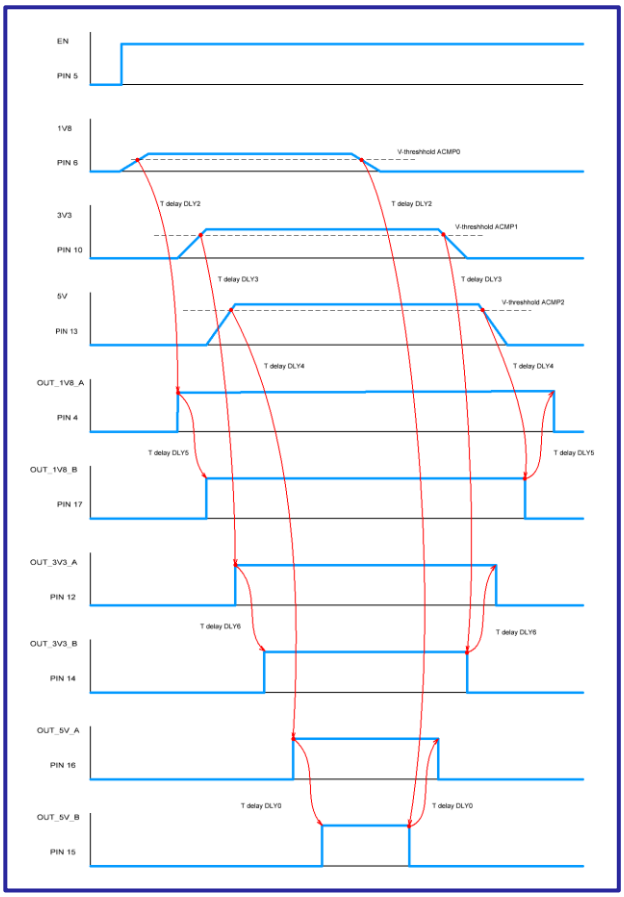


所需元器件

- 任何带有 3 个模拟比较器的 GreenPAK IC(CMIC)
- 6 个电阻

GreenPAK 设计图



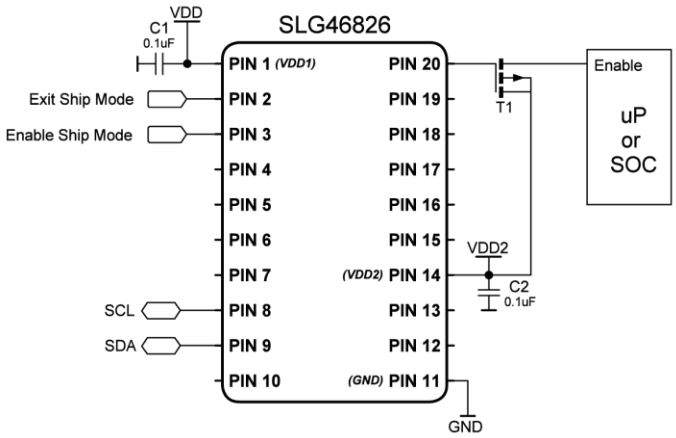


设计步骤

1. 将 EN 和 Voltage Monitoring 的 PINs 脚配置为输入。
2. 将系统电源控制时序信号的 PINs 脚配置为输出。
3. 启动 ACMP, 将 POR 连接到 **PWR UP**， 并将每个 ACMP 的 **IN-** 端配置到所需的电压阈值（比较器反转电压阈值）。
4. 根据需要的延时时间配置 DLY 的 **Counter data**。
5. 根据需要的逻辑功能配置 LUT 。

应用：运输模式控制器

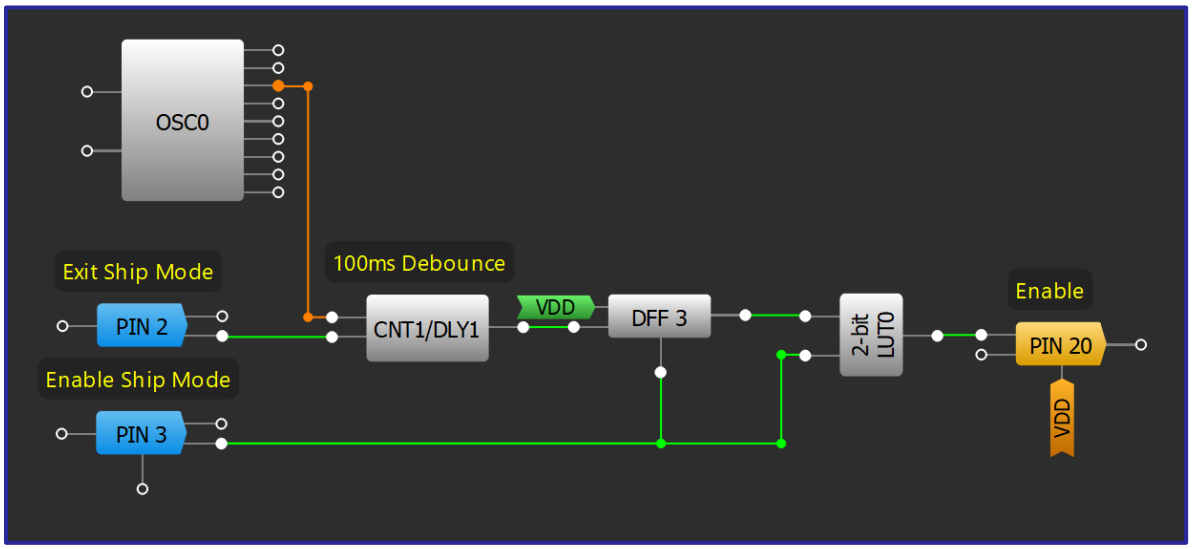
在产品还未交付到用户手上之前，使用一个超低功耗的按钮监视器能够延长电池放电时间，可以给用户提供一个更好的上手体验。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 外部 PMOS 负载开关

GreenPAK 设计图

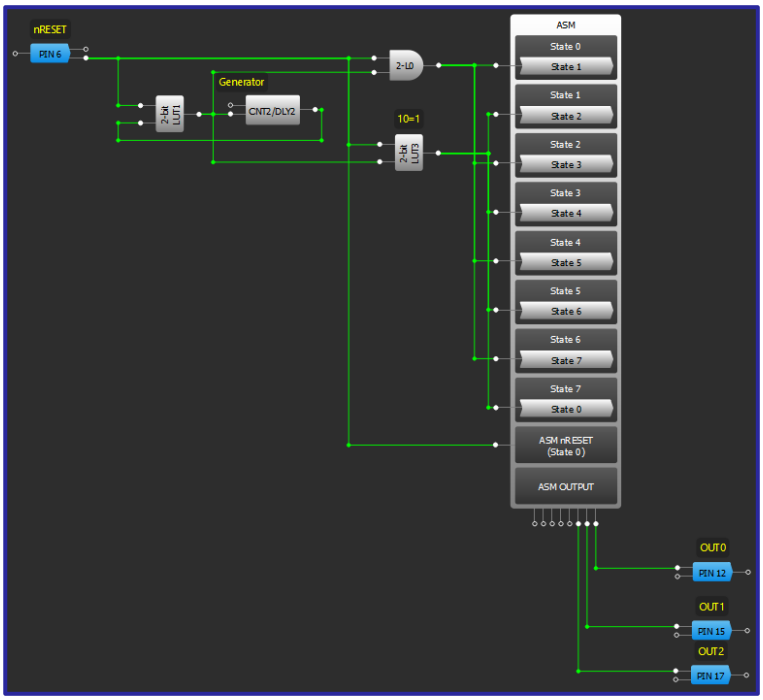


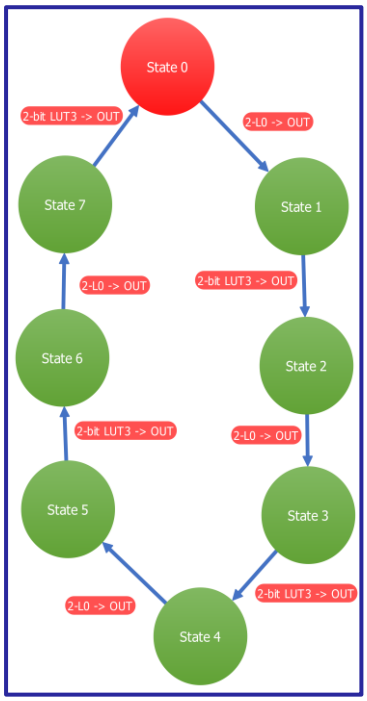
设计步骤

1. 配置 PIN2 为带 1MΩ 上拉电阻的输入引脚。
2. 根据需要的去抖动时间配置 **CNT1/DLY1** 的 **Counter data**。
3. 根据进入运输模式和退出运输模式对应的输出电平配置 LUT 的逻辑。

技巧：由异步状态机创建同步状态机

同步状态机 (SSM) 是在转换条件满足的情况下在时钟沿到达时进行状态转移。把 GreenPAK 芯片中的 **异步状态机（ASM）**宏单元转换成同步状态机的通常方法是使用脉冲宽度大于 ASM 状态转换时间的时钟信号。





上图展示了 SSM 在 3 位计数器中的应用。 **CNT2** 和 **2-bit LUT1** 用于生成时钟信号。 ASM 的 8 个状态位串联连接。 **2-bit LUT0** 和 **2-bit LUT3** 用于防止一个逻辑高电平信号使两个相邻状态位同时发生状态转换。 ASM 输出的状态值如下表所示。

RAM

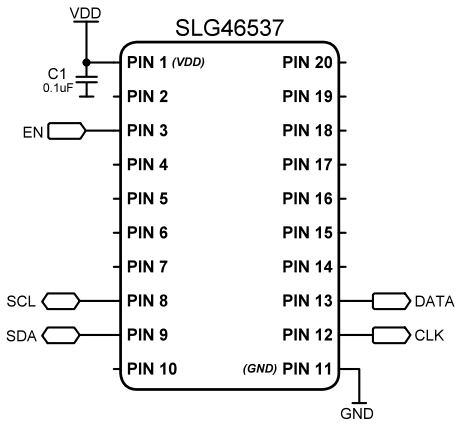
State name	Connection Matrix Output RAM							
	OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
State 0	0	0	0	0	0	0	0	0
State 1	0	0	0	0	0	0	0	1
State 2	0	0	0	0	0	0	1	0
State 3	0	0	0	0	0	0	1	1
State 4	0	0	0	0	0	1	0	0
State 5	0	0	0	0	0	1	0	1
State 6	0	0	0	0	0	1	1	0
State 7	0	0	0	0	0	1	1	1

当 **6 脚**变为高电平时，ASM 由复位状态 (State 0) 转移为下一状态 (State 1)。 以下状态转换是随着时钟信号 **CNT2** 不断切换为高低电平而完成的。

更多关于异步状态机转换为同步状态机的信息，请参考 “AN-1126 ASM 到同步状态机转换”

应用：N 比特流

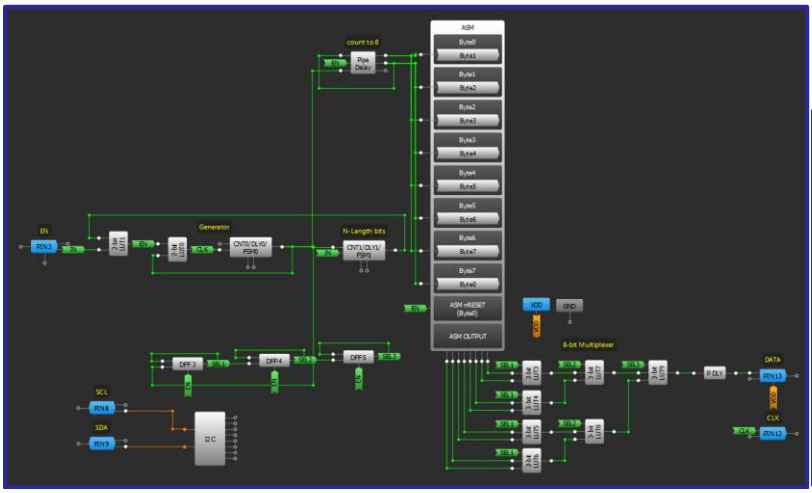
比特流是在通信路径上连续传输的比特序列。GreenPAK 能够创建多达 64 位的重复字符串。

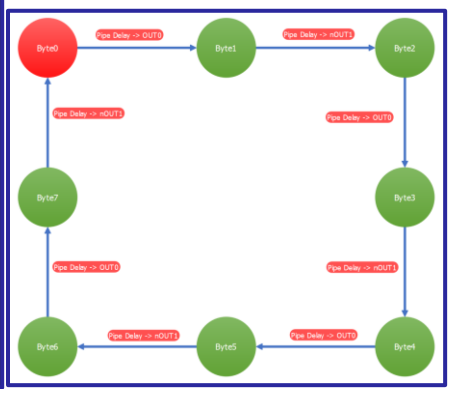


所需元器件

- 任意带有 ASM 模块的 GreenPAK IC(CMIC)

GreenPAK 设计图





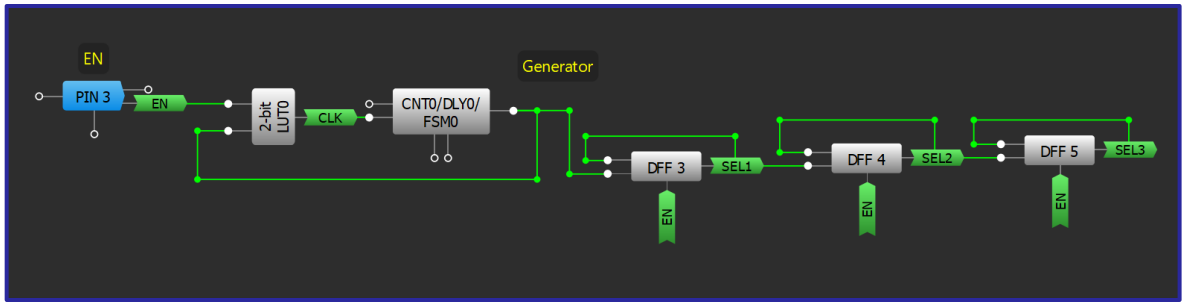
设计步骤

1. 使用 应用：8 位多路复用器 配置信号发生器和 8 位多路复用器。
2. 配置 **CNT1** 定义比特流的长度。
3. 使用 技巧：由异步状态机创建同步状态机 配置 ASM。
4. 将 8 位多路复用器和信号发生器的输出连接至输出脚。
5. 可以通过 I²C 改变比特流的长度 (**CNT1** 的 counter data) 。
6. 也可以通过 I²C 改变 ASM 输出 RAM 中的数据。

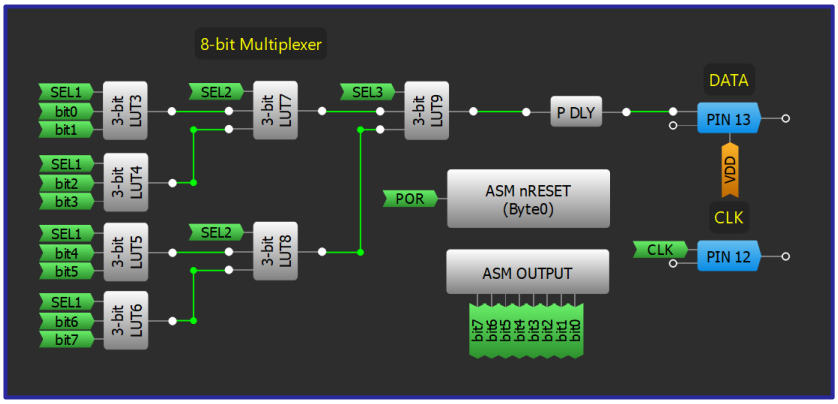
技巧：多路复用比特流

此技巧适用于所有GreenPAK 芯片。

GreenPAK 常用于数据传输。 通过 GreenPAK 传输数据 或 从 SoC 传输多路数据， 必须将数据合并从而在一条线路上进行传输。以下是 GreenPAK 传输源自 ASM 输出 RAM 的多路数据的一个示例。



发生电路如上图所示。由 **EN** 信号控制，发生器可实现由 CLK 产生，可实现数据的同步传输。发生器设定了将传输数据复合到一条线上的操作规则。

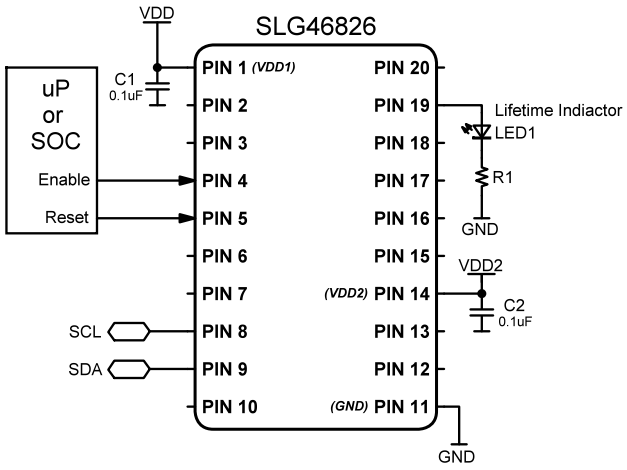


IN3	IN2	IN1	IN0	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

8 位多路复用器如上图所示。 将 7 个 3-bit LUT 配置为 8 位多路复用器 (参考 AN-1003)。 **多路复用[MUX]真值表**如右上所示。多路复用器按照发生器设定的复合规则将 ASM 模块输出的数据组合在一起后传输到 **DATA**。当 **EN** 为低电平时，**DATA** 的输出结果与 ASM 输出 RAM 的最高有效位一致。ASM 输出 RAM 可以通过 I²C 改变，也可以通过触发 ASM 的状态改变来改变数据的各个位。 如果 ASM 不可用也可以通过手动将输入拉高或拉低来修改数据位。

应用：10 年定时器

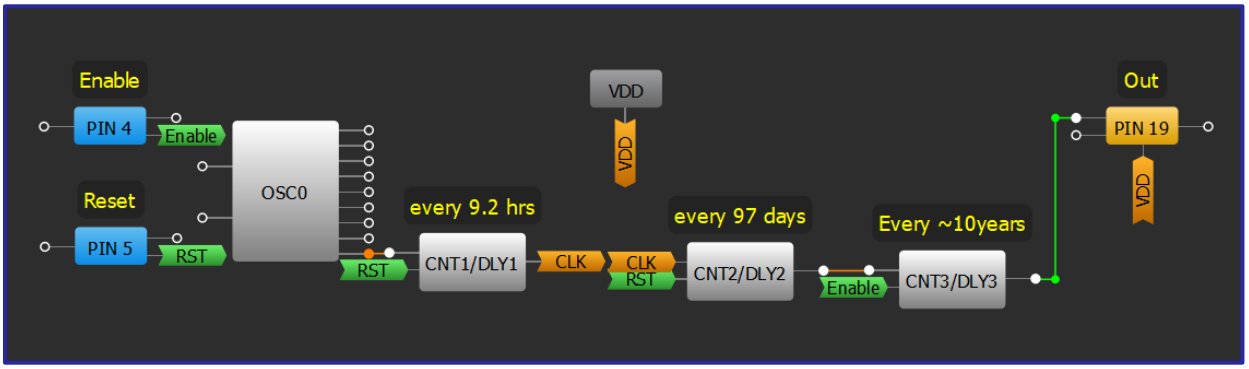
超长定时器用于确定产品的使用寿命，而无需占用太多功耗预算。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

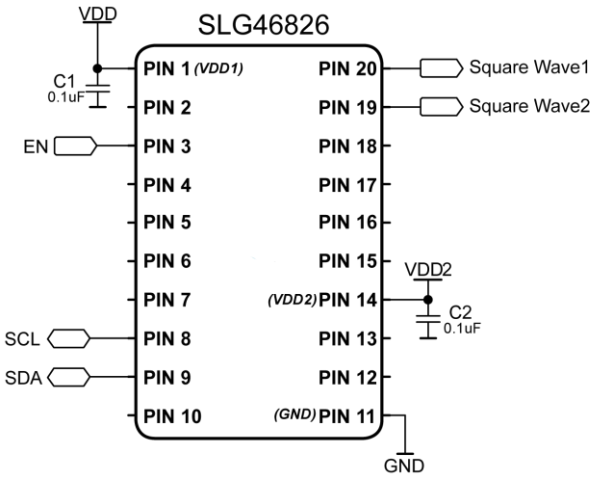


设计步骤

1. 使用 技巧：CNT/DLY 的级联 将各个 Counter 模块级联。
2. 连接输入引脚和输出引脚。
3. 配置 CNT 的 Properties 。

应用：方波发生器

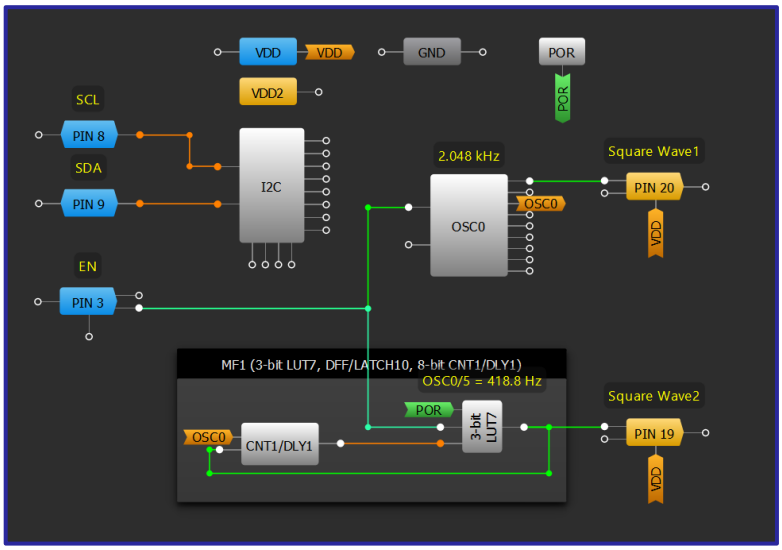
方波对于数字时钟系统是必不可少的。 它们可以通过使用振荡器块或定制频率的延迟逻辑，从而很容易地在 GreenPAK 中实现。



所需元器件

- 任意 GreenPAK IC
- 无需其他元器件

GreenPAK 设计图



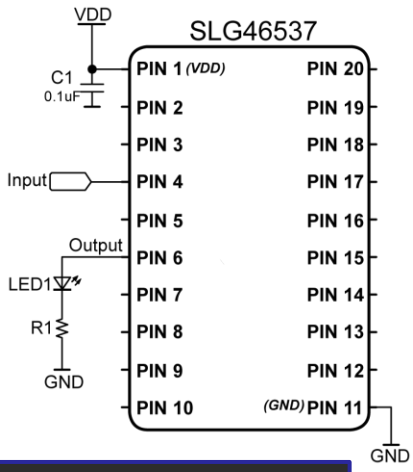
设计步骤

1. 配置 **EN** 输入和方波输出。
2. 使用内部振荡器在 **PIN20** 上产生一个方波。 **“CLK” 预分频器**和 **“OUT0” 二次分频器**可以更改以自定义频率。
3. 使用边缘延迟(CNT1)和 LUT 在 **PIN19** 上产生一个方波。 这种配置允许用户通过更精细的调整来分割方波的频率。

$$Division\ Coefficient = Counter\ Data + 2$$

应用：按下多个事件按钮

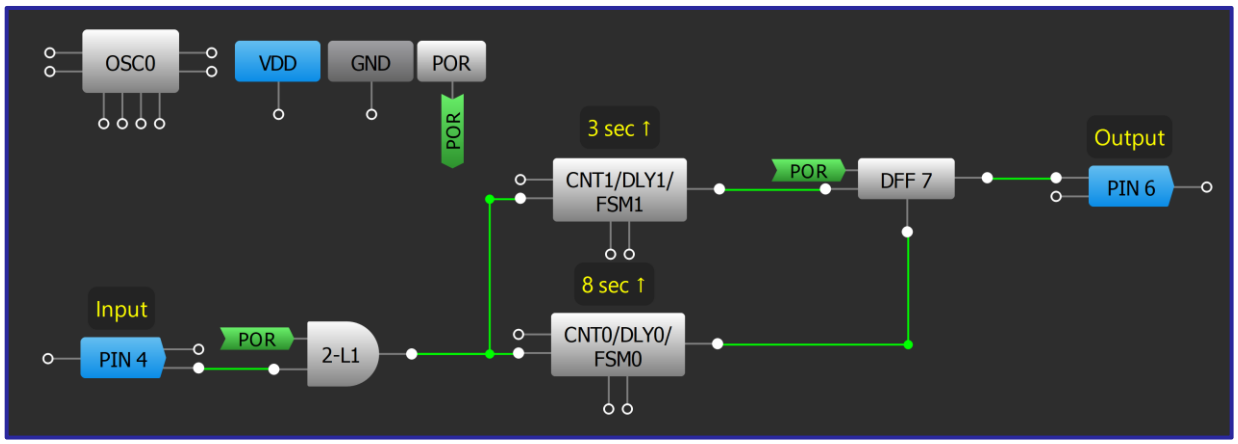
使用单个按钮生成多个事件是节省外部控制组件的常见解决方案。 使用一个按钮和预定的时间间隔，您可自定义控制一个 LED 手电筒。



所需元器件

- 任意 Green PAK IC
- 1 个 LED
- 1 个电阻器

GreenPAK 设计图



设计步骤

1. 将 GPIO 引脚配置为按钮的输入和 LED 控制的输出。
2. 添加 **CNT0/DLY0**, **CNT1/DLY1** 和一个 DFF 来记住最后的状态。
3. 将 CNT/DLY 模块配置为“延迟”模式，**Edge select 选**为“Rising”。
4. 将 **CNT0/DLY0 的**输出配置为“nOUT”。

第 3 章:信号调理

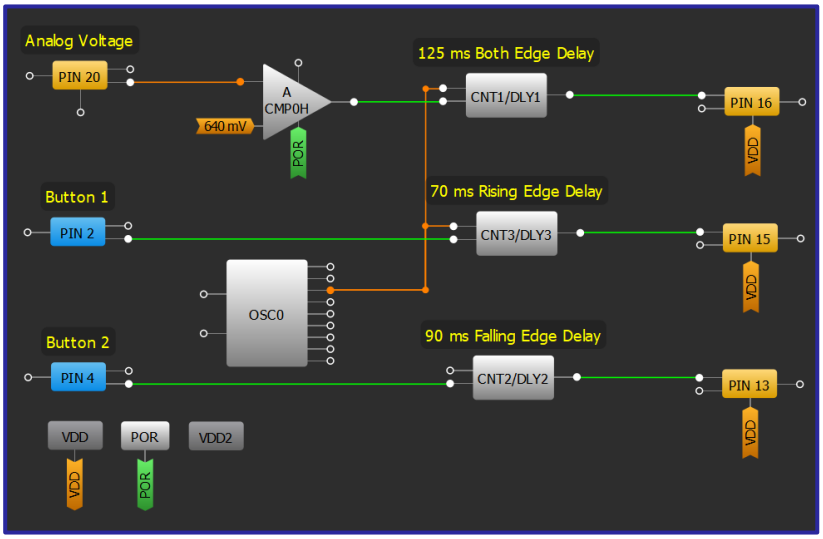
本章通过详细讲解外部信号并使其成为对系统操作有用的应用。 涉及到这方面的一些应用包括分频/倍增、滤波器和传感器控制器。

技巧：利用 CNT/DLY 滤除毛刺

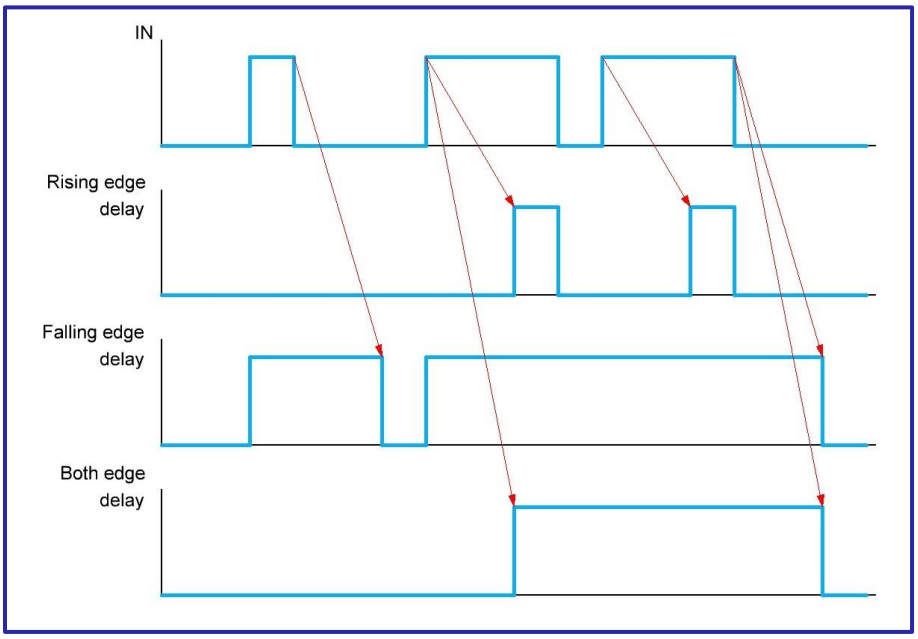
此技巧适用于所有 GreenPAK 芯片。

去毛刺/去抖动过滤器用于消除毛刺[虚假的信号暂态]。毛刺可以在很多情况下产生，比 如，按钮的按压、释放或电压非常接近不带迟滞的引脚阈值的时候。

有 3 种可选的边沿触发选项来抑制毛刺：上升沿、下降沿和双边沿，Delay 将会滤除短于 Delay 值的脉冲，3 种不同配置的影响见下图：



防抖延时示例

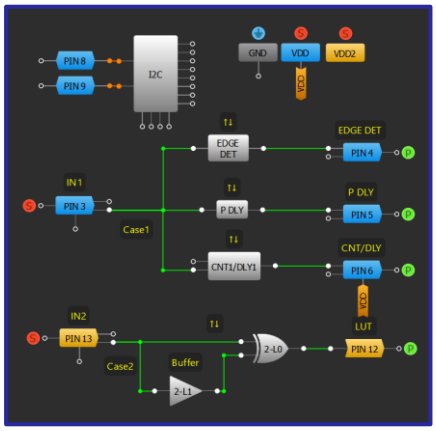


不同边沿延时时序

技巧：边沿检测器

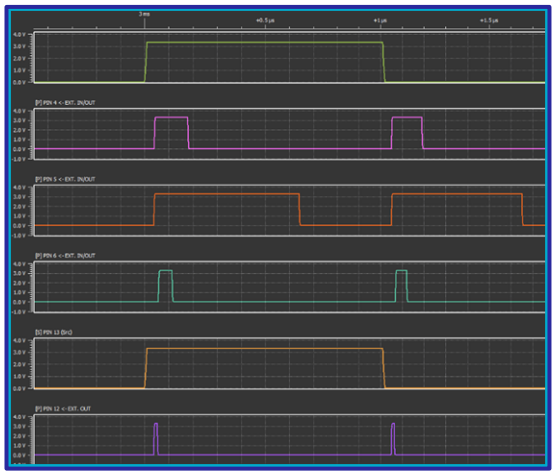
此技巧适用于所有GreenPAK 芯片。

边沿检测器在数字电路中是非常重要的组件。它是一种单路输入和单路输出的简单电路。边沿检测器在检测到边沿信号时（上升沿，下降沿，或者两者兼具的双边沿）产生一个短的脉冲信号。通常用于实现复位功能、看门狗定时器或者其他需要边沿触发的应用。有以下几种方法实现边沿检测（见下图）。了解更详细的如何建立一个边沿触发器以及更多的应用案例请参考 AN-1046 多种边沿检测电路。



边沿检测器示例

方法 1 使用边沿检测(EDGE DET)、可设定延时器(P DLY)和计数器/延时器(CNT/DLY)模块来获得边沿检测器。与方法 2（如下图）相比，这些模块可产生一个持续时间更长的脉冲。

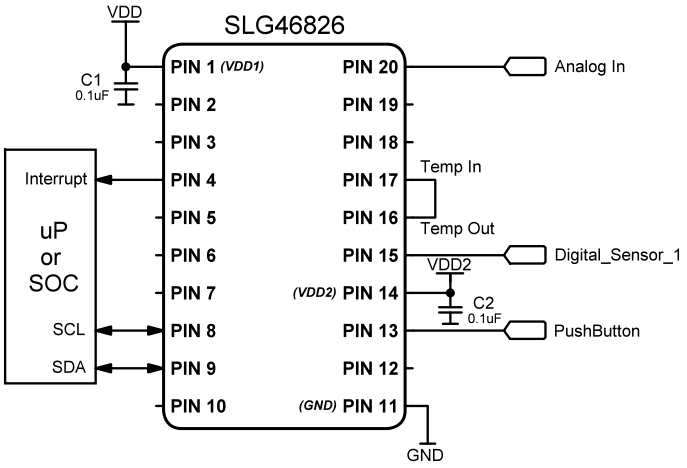


不同的边沿检测器时序

方法 2 利用信号经由缓冲器传输的短暂延迟。这个延迟信号通过一个 2 位的 XOR（异或）与初始输入信号进行逻辑异或，原始输入信号的传输时间非常短。通过缓冲器的短暂延迟会导致 XOR 输入之间的差异，从而在其输出上生成一个短脉冲。由于查找表的内部结构，它们的每个输入具有不同的传输延迟时间。

应用：中断控制器

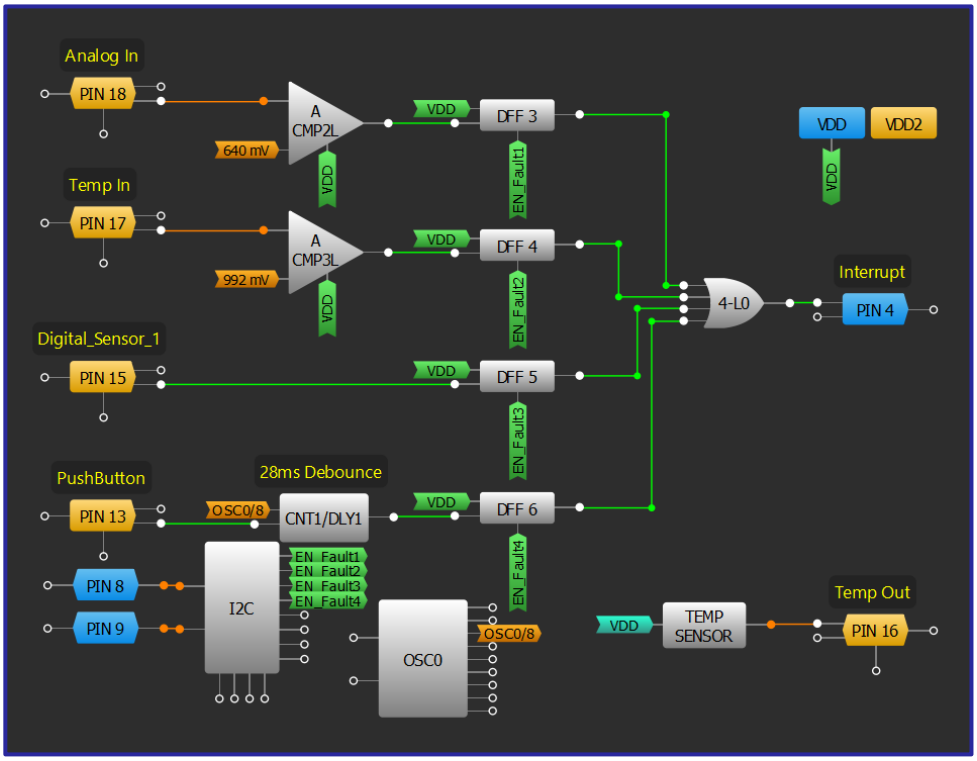
GreenPAK 可以配置为监控多个不同中断信号，并将这些不同的中断信号整合为一个中断信号以通知处理器进行处理，微处理器或者 SOC 可以通过 I²C 接口读取每个 DFF 的输出值来确定中断源。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



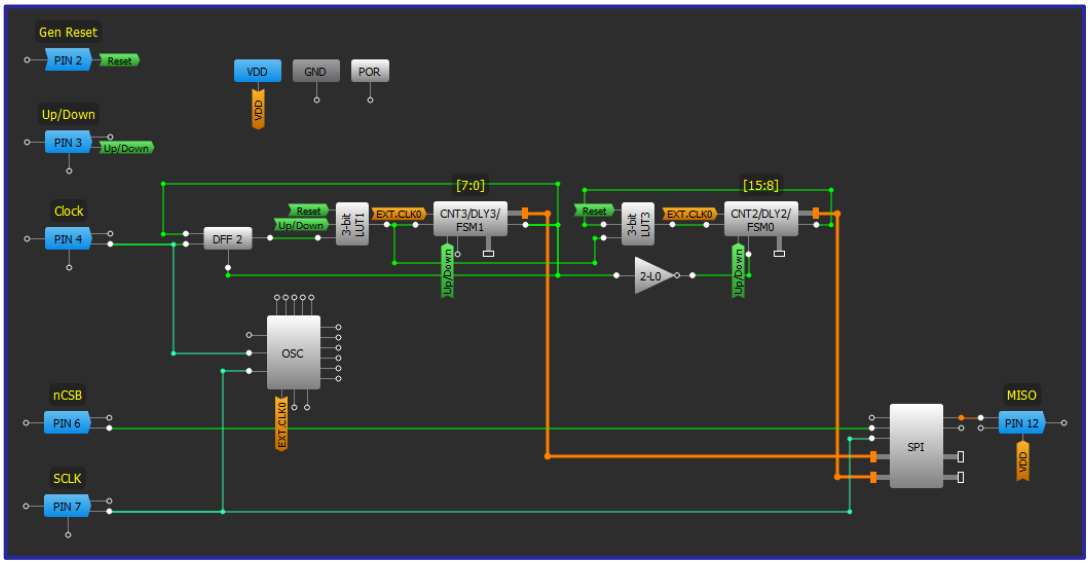
设计步骤

1. 配置 **PIN16**、**PIN17**、**PIN18** 为 **Analog Input/Output**。
2. 根据需求为 ACMP 配置合适的阈值。
3. 将所有 DFF 的输出连接到 OR 门，并将 **PIN4** 配置为 **Digital output**。
4. 根据需要的去抖时间配置 **DLY1** 的 **Counter data**。

技巧：构建一个双向计数器

此技巧适用于所有带有SPI 接口的GreenPAK 芯片。 另外一种方法是使用其他带有FSM 模块的GreenPAK 芯片，使用I²C 读取命令、并行输出或其他方式存储计数器信息。

计数器是用于记录事件（脉冲、边沿）数量的基本电路，它主要由具有记忆功能的触发器构成。 GreenPAK 芯片中，CNT/DLY 模块的功能非常强大，可用作有限状态机（FSM），进行向上计数、向下计数或保持当前值，具体由 GreenPAK 电路的连接方式而定。此技巧通过使用 GreenPAK 芯片中的 2 个 FSM 模块，监测脉冲输入（**时钟**）数量和利用 SLG46140V 的 SPI 宏单元输出 16 位脉冲序列。



带有 SPI 输出 的 16 位 FSM

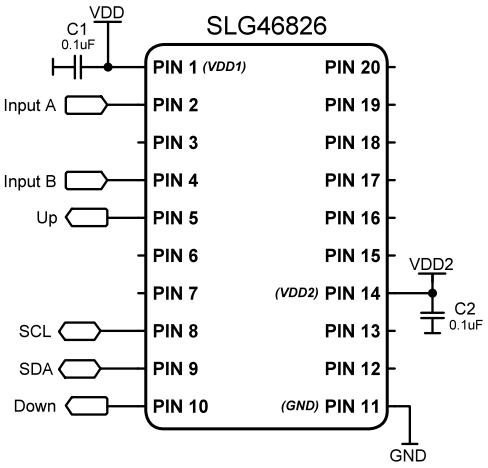
“16 位 FSM 输出连接至 SPI”通过一个 16 位寄存器 (**FSM0, FSM1**) 对输入时钟脉冲计数。 用户可以随时通过 SPI 读取该数值，复位 16 位寄存器或者改变计数方式。

该 16 位计数器使用 2 个计数模块 (**FSM0** 和 **FSM1** 模块) 和一些额外的逻辑单元。 高位 [15:8] 存储在 **FSM0**，低位 [7:0] 存储在 **FSM1**。两个 FSM 均连接到 SPI 模块转换成串行数据输出。计数模式由连接到 FSM 模块的 Up/Down 脚控制，直接连接到 FSM 模块的 **UP** 脚。 当该脚为高电平时，系统向上计数；当该脚为低电平时，系统向下计数。 **Gen Reset** 脚用于复位两个计数器的值（高电平有效）。

时钟输入脚同时连接到 **FSM1** 和 **FSM0** 的 **CLK**。 **FSM1** 对每个时钟周期进行计数；当 **FSM1** 的数值达到计数上限值 255 且 Up/Down 信号为高电平或者当 **FSM1** 的数值达到计数下限值 0 且 Up/Down 信号为低电平时，**FSM0** 开始进行计数。 上述控制通过 FSM0 的 **Keep** 脚实现。 当该信号为高电平时， 无论是否有时钟信号输入，**FSM0** 的计数值不变。 **FSM1** 的输出信号通过一个反相器连接到 FSM0 的 **KEEP** 脚。 **FSM1** 计数时，输出信号为低电平；只有当 **FSM1** 的数值达到计数上限值 255 且 Up/Down 信号为高电平或者当 **FSM1** 的数值达到计数下限值 0 且 Up/Down 信号为低电平时，**FSM1** 输出信号为高电平。

应用：编码器

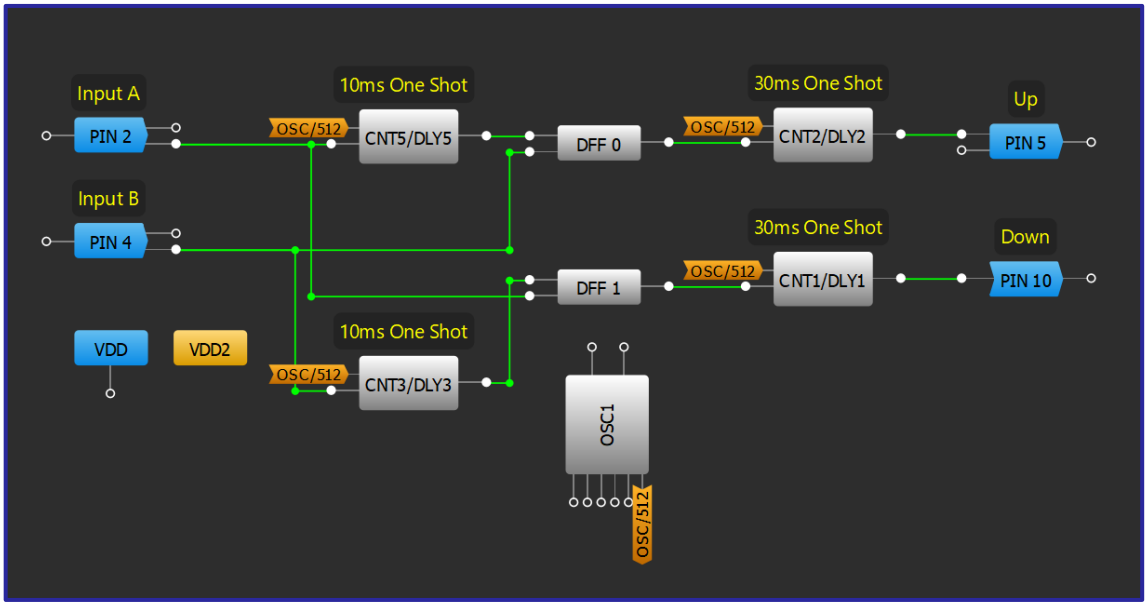
编码器用来将旋转运动或者线性运动转换为数字信号，下面的设计非常适合鼠标滚轮和耳机音量控制这类应用。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

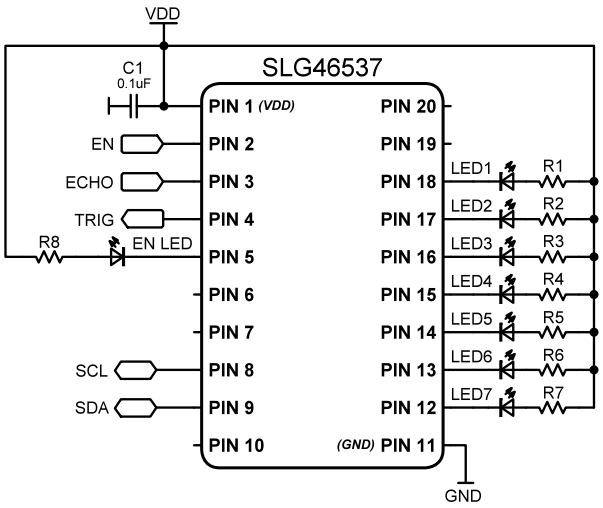


设计步骤

1. 将 A 和 B 信号输入脚配置为 **Digital input**。
2. 将两个引脚配置为输出来指定编码器的运动方向。
3. 根据滤波需要，将 **CNT3/DLY3** 和 **CNT5/DLY5** 配置为 **One shot** 模式并设置相应的 **Counter data**。
4. 配置 DFF 以检测运动方向 (上或下)。
5. 根据需要输出的脉冲宽度，将 **CNT1/DLY1** 和 **CNT2/DLY2** 配置为 **One shot** 模式，并设置相应的 **Counter data**。

应用：距离感应器

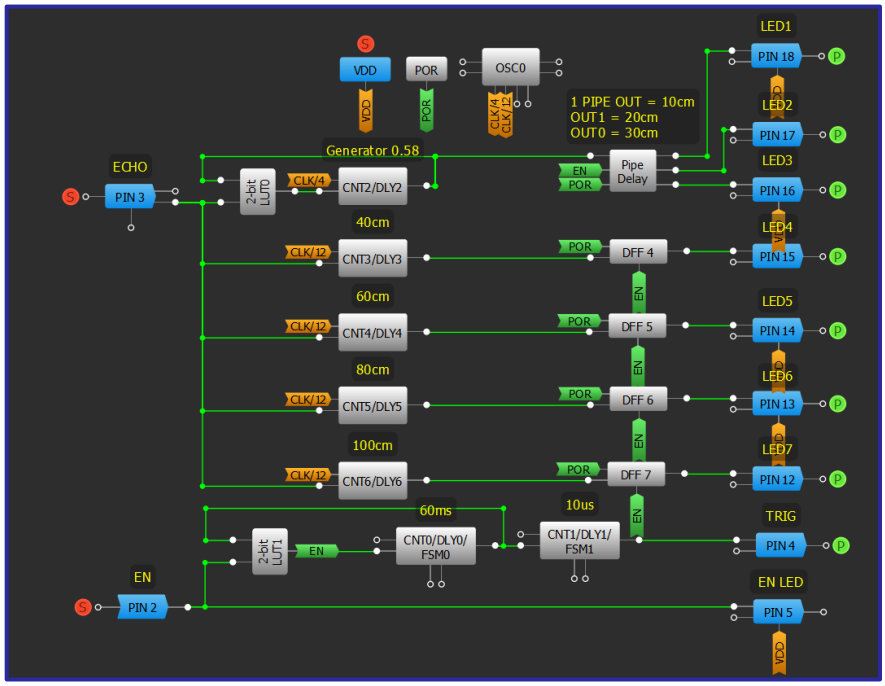
超声波测距模块可提供非接触式距离感测功能。本设计是一种基于HC-SR04 的超声波测距仪控制器。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 每个距离测量所需的 LED
- 每个距离测量所需的电阻

GreenPAK 设计图

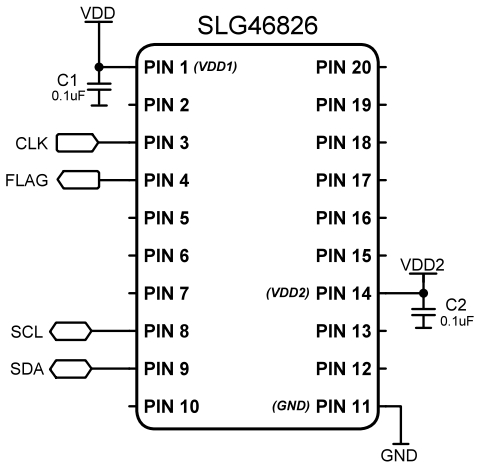


设计步骤

1. 配置 **ECHO** 对应的 I/O 为 Digital input, **TRIG** 对应的 I/O 为 Digital output。
2. 添加 LUT 逻辑单元和 **CNT/DLY0** 以创建一个带有 ENABLE 信号的发生器。
3. 添加 Pipe Delay 和 **CNT/DLY2** 以创建一个距离探测发生器。
4. 将 CNT/DLY 模块配置为 **rising edge delay** 模式以测量不同的距离。
5. 添加并配置 DFFs 来保存距离信息。
6. 将各个 DFF 模块的 OUT 连接到相应的输出引脚并配置为 **open drain**。

应用：频率范围探测器

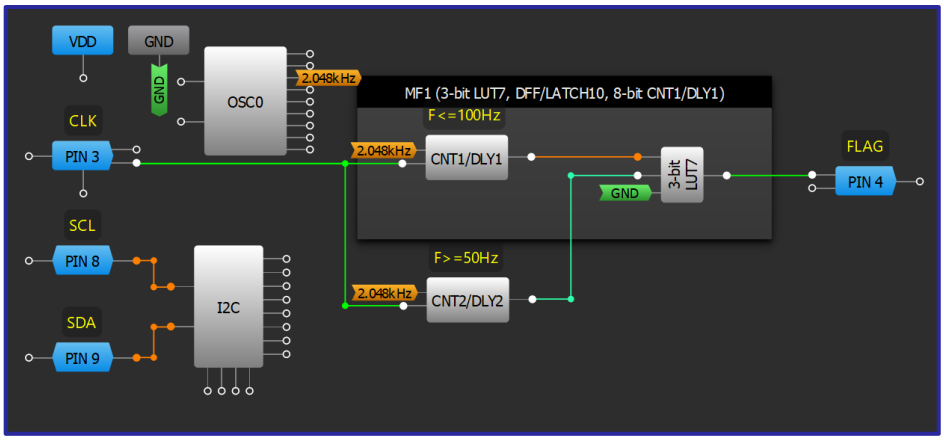
许多设备都有特定的工作频率范围，这要求它们运行并要求对输入时钟信号进行检测，确保此时钟频率符合要求。本设计可用于检测输入时钟频率是否在所需的频率范围内。

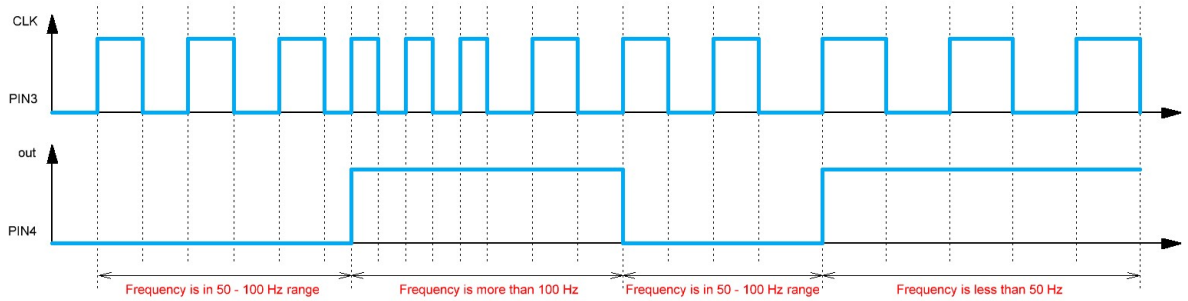


所需元器件

- 任意 GreenPAK IC(CMIC)

GreenPAK 设计图



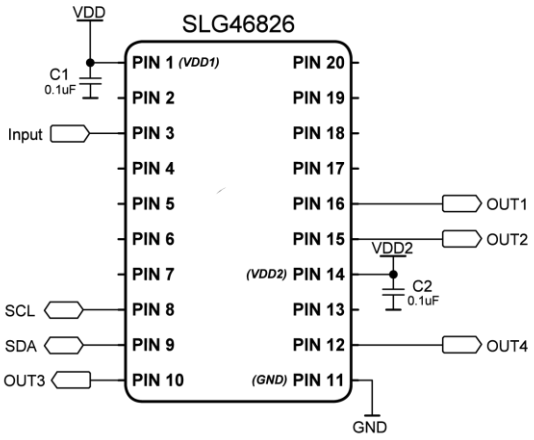


设计步骤

1. 配置时钟信号对应的 I/O 为 Digital input, 标志位对应的 I/O 为 Digital output 。
2. 将 CNT/DLY 模块配置为 **Frequency detect** 模式，Edge detect 选为 **rising** 。
3. 将每个 CNT/DLY 模块分别设置为最小和最大频率值。
4. 把一个 LUT 配置成如果频率在所需范围之外将会输出高。

应用：分频器

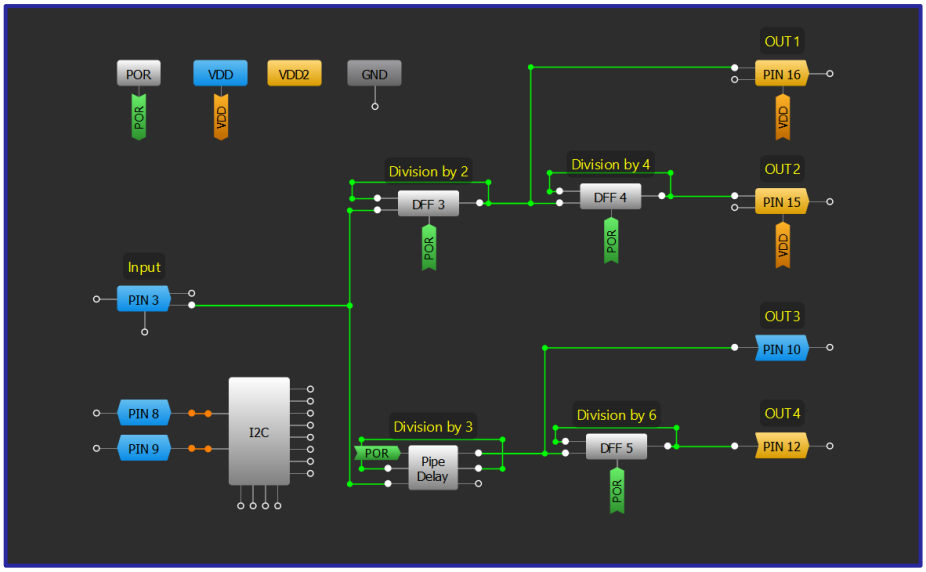
分频器用于按不同的系数降低输入信号源频率，可用于提高电子对抗设备、通信系统和实验室仪器的性能。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



设计步骤

1. 配置信号输入源对应的 I/O 为 **Digital input**
2. 利用 DFF 和 Pipe Delay 分割一次频率。
3. 利用另外一个 DFF 将信号频率又分割一次。
4. 为各个输出添加并配置 LUT。

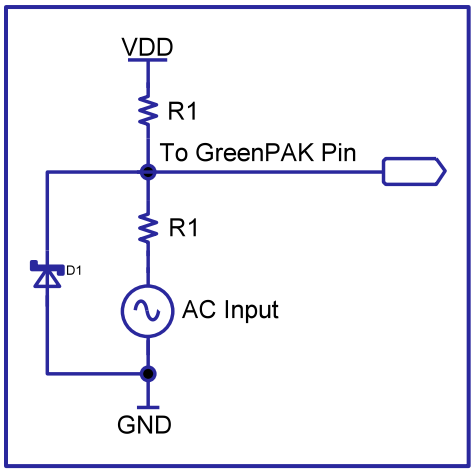
技巧：过零检测

此技巧适用于包含 ACMP 的 GreenPAK。

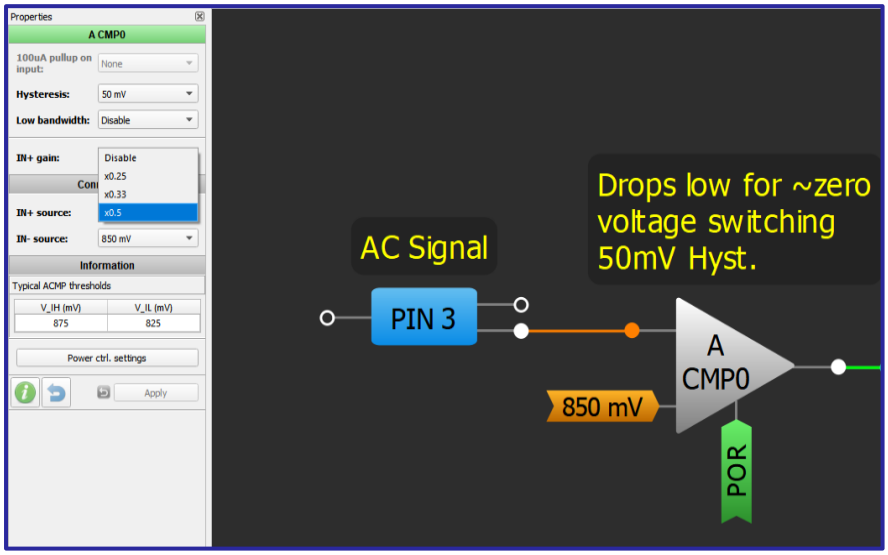
过零检测通常应用于检测准确的交流特性，比如频率和相位。

GreenPAK 的输入电压范围为 0V 到 VDD，VDD 最大为 5.5V，利用 GreenPAK 检测一个交流信号的过零点，需要先将一个直流偏置信号叠加到交流信号上，如 右图 - **基本的直流偏置电路** 所示的那样，通过在 VDD 和交流信号中间添加一个 1:1 的分压电阻来实现。

过零检测至少需要一个或者两个比较器和一个计数器，比较器根据参考电压检测交流输入信号，参考电压既可以是 GreenPAK 内部的参考电压也可以是外部参考电压， 如果所需的过零检测阈值电压大于 GreenPAK 内部所有的参考电压，可通过在 Properties 窗口中设置 **IN+ gain** 以降低交流信号幅值。



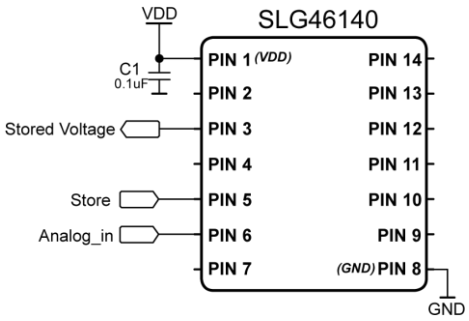
基本的直流偏置电路



利用 IN+ 增益进行 1.7V 阈值检测

应用：模拟存储元件

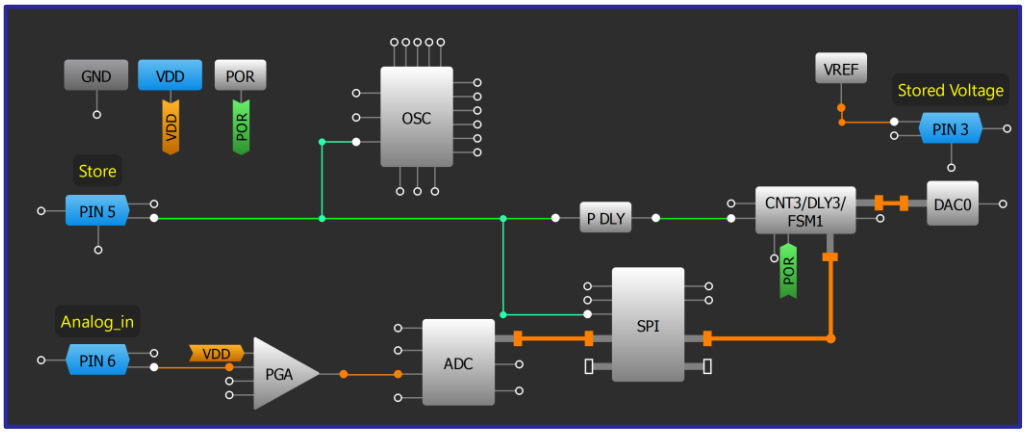
该应用可用于在输出上存储模拟电压，直到上升沿被应用到 Store 输入。 输入输出模拟电压在 0-1V 范围内。

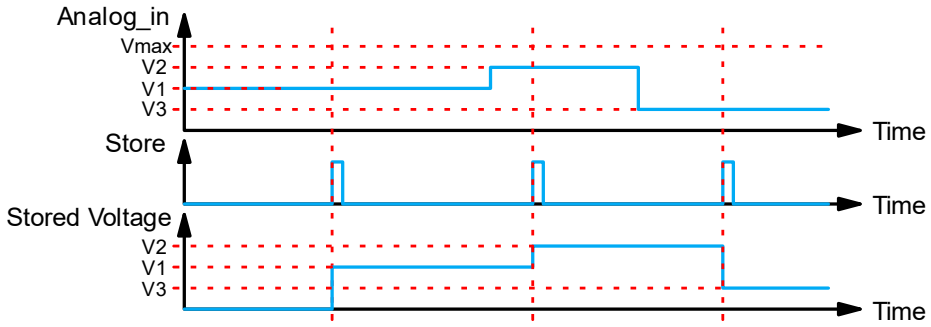


所需元器件

- 任意带有 ADC, SPI, 和 DAC 模块的 GreenPAK IC

GreenPAK 设计图





设计步骤

1. 配置 SPI 为“ADC/FSM buffer”模式，修改 **PAR input data source** 为“ADC”。
2. 配置 **FSM0** 为“Set (counter value = FSM data)”，修改 **FSM data source** 为“SPI[7:0]”。
3. 配置 DAC **Input selection** 为“From DCMP1’s input”，VREF **Source selector** 为“DAC0 out”。
4. 将 Store 输入直接连接到 SPI **SCLK**，通过 P DLY (设置为)边缘延迟后连到 FSM1 **SET IN**。

第 4 章:安全功能

本章介绍了一些应用程序，这些应用程序旨在响应系统中的故障条件，并保护系统免受损坏。 一些为电子系统提供安全性的应用是指充电量示器、看门狗定时器和温度传感器。

技巧：如何降低 ACMP 的功耗

此技巧适用于包含 ACMP 的 GreenPAK，但是不同芯片功耗降低的多少会有所不同。

GreenPAK 通常用来降低系统功耗，然而 GreenPAK 内有几种组件在启用时会显著增大功耗，其中最明显的就是模拟比较器，表 1 截取自 SLG46826 的数据手册，并用红框突出显示了 ACMP 在各种条件下的电流消耗值。

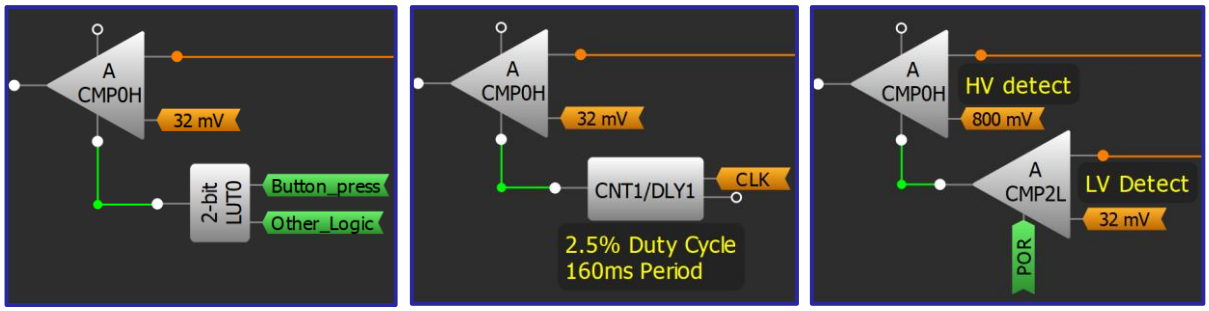
表 1 SLG46826 电流消耗值

Note	V_{DD} = 2.5 V	V_{DD} = 3.3 V	V_{DD} = 5.0 V	Unit
Chip Quiescent	0.39	0.43	0.53	µA
Vref OUT0 (Source none, Source Temp Sensor, Buffer On)	12.79	12.95	13.57	µA
Vref OUT0 (Source none, Source Temp Sensor, Buffer Off)	7.62	7.67	7.87	µA
Vref OUT1 (Source none, Buffer On)	6.53	6.61	7.02	µA
Vref OUT1 (Source none, Buffer Off)	1.40	1.44	1.54	µA
Vref (ACMPxH, 0.32 mV, Buffer On)	12.24	12.59	12.21	µA
Vref (ACMPxL, 0.32 mV, Buffer On)	6.93	7.01	7.43	µA
ACMP0H, 1H, 2L, 3L,hysteresis disabled, gain = 1, +IN - IO11, 12, 13, 14 Pull Up 1M, Vref = 32 mV	65.86	67.12	70.77	µA
ACMP0H, 1H, 2L, 3L,hysteresis disabled, gain = 1, +IN - IO11, 12, 13, 14 Pull Down 1M, Vref = 32 mV	37.34	38.05	40.29	µA
ACMP0H, 1H, 100 µA disabled, hysteresis disabled, gain = 1, +IN - IO13, 14 Pull Up 1M	63.85	65.11	68.71	µA
ACMP0H, 1H, 100 µA disabled, hysteresis disabled, gain = 1, +IN - IO13, 14 Pull Down 1M	35.97	36.68	38.87	µA
ACMP0H, 100 µA disabled, hysteresis disabled, gain = 1, +IN - VDD, Vref = 32 mV	36.30	36.96	38.85	µA
ACMP0H, 100 µA enabled, hysteresis disabled, gain = 1, +IN - IO14 Pull Up 1M, Vref = 32 mV	46.77	47.31	49.23	µA
ACMP0H, 100 µA enabled, hysteresis disabled, gain = 1, +IN - IO14 Pull Down 1M, Vref = 32 mV	49.02	50.29	53.75	µA

幸运的是，ACMP 可以在不使用时关闭，这可以通过 2 种方式来实现：

1. 通过控制 ACMP 的 **PWR UP** 脚。
2. 为 ACMP 启用唤醒-休眠定时器(**WS Ctrl**)。

唤醒-休眠控制需要将一个专用的定时器配置在 “Wake sleep controller” 模式，许多(但不是所有) GreenPAK 都包含此功能。 **PWR UP** 的控制方式可以应用于所有带 ACMP 的 GreenPAK 芯片中， 当信号为高时 ACMP 启用，通过使用逻辑、计数器或者其他组件去关闭 ACMP，这样可以显著降低功耗，例如如果需要 2 个不同的电压比较阈值，则高阈值的比较器可以只等低阈值的比较器达到阈值之后，才启用。



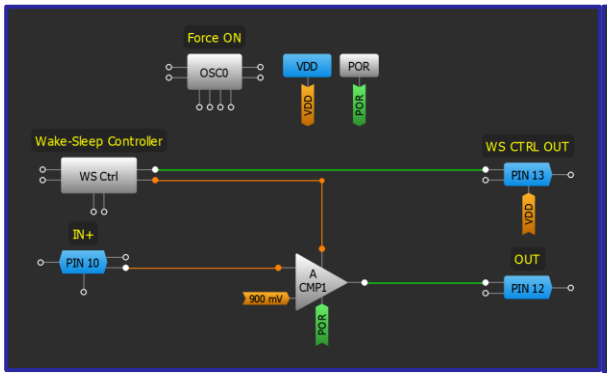
(a) Logic (b) Duty Cycle (c) Cascaded

常见的 PWR UP 配置

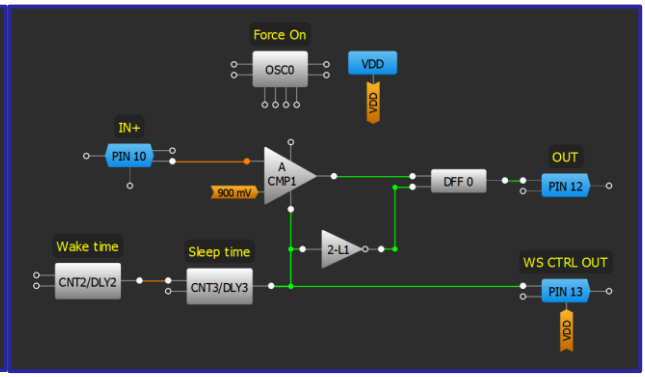
技巧：唤醒与休眠

让宏单元进行休眠与唤醒可以有效降低系统功耗。模拟宏单元如ACMP 和ADC 均可进行唤醒-休眠控制。

唤醒-休眠控制，允许周期性地打开和关闭这些宏单元。对于带有 **WS Ctrl** 模块的 GreenPAK 芯片，可利用 WS Ctrl 模块实现唤醒-休眠控制。 对于没含有此模块的 GreenPAK 芯片， 可以使用 2 个计数器，1 个 D 类触发器和 1 个反相器来实现该功能。下图中展示了上述方法的实例。



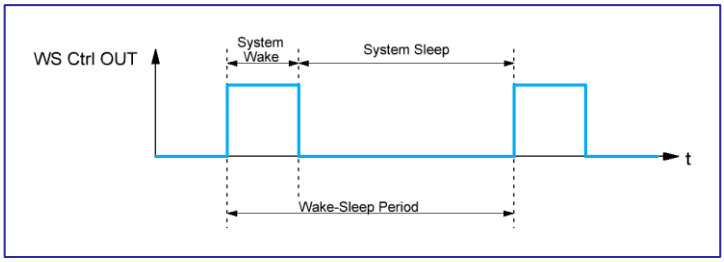
WS Ctrl 方式



2 个计数器 方式

没有唤醒/休眠功能时，电路消耗的总电流由以下两部分构成：

- 静态电流
- ACMP 电流



唤醒-休眠 时序

增加唤醒/休眠控制后，静态电流近似计算如下：

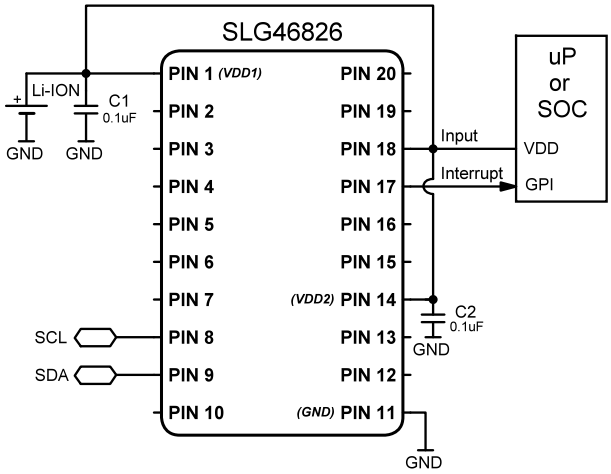
$$I_{WS} = \frac{System\ Wake}{System\ Wake + System\ Sleep} * I_{without\ WS} = \frac{System\ Wake}{WS\ Period} * I_{without\ WS}$$

带有唤醒/休眠功能时，电路的总电流为：

$$Total\ Current = I_{Quiescent} + I_{OSC} + I_{Wake\ Sleep}$$

应用：窗口比较器

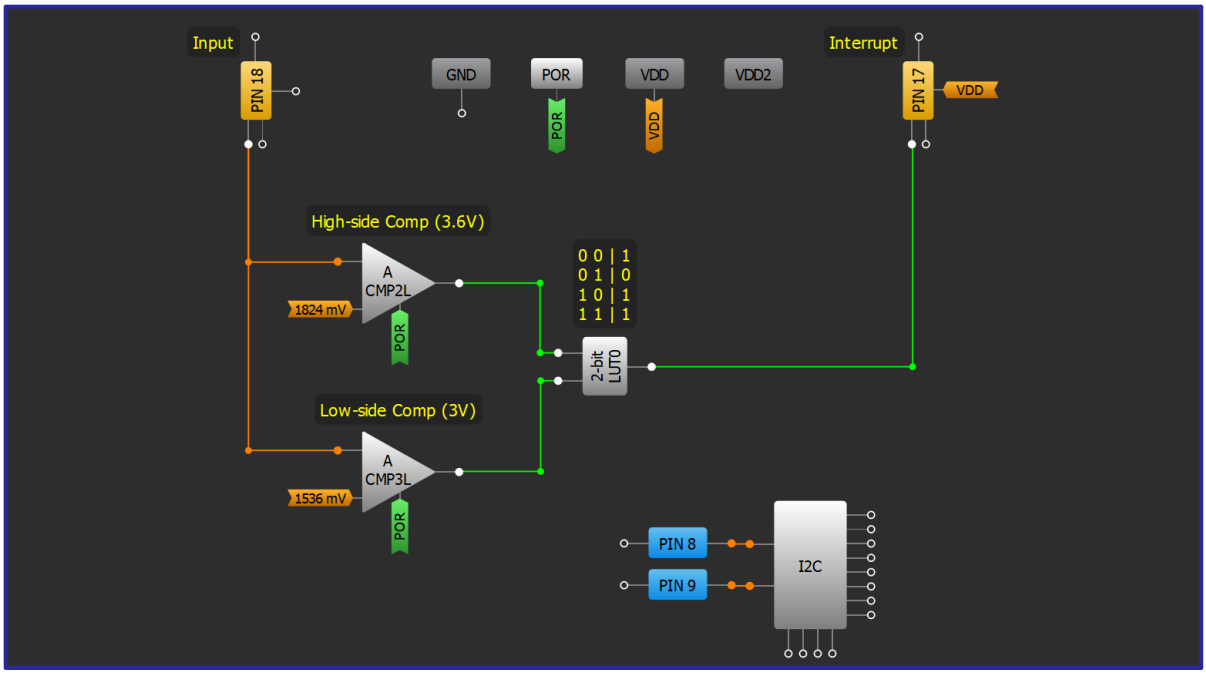
在一个利用电池或者超级电容等存储型电源来供电的设计中，窗口比较器是必须的，通过监测电池电压，设备可以在低电池电量下停止使用不必要的资源，以防止造成设备的永久性损坏。



所需元器件

- 任意带有 ACMP 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

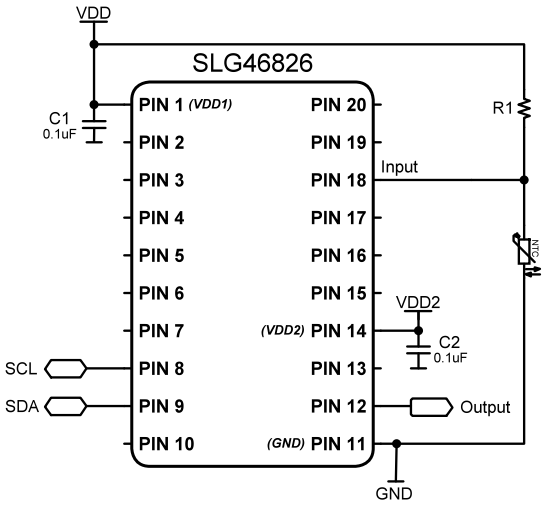


设计步骤

1. 配置比较器 **ACMP2L** 的 **IN- source** 和 **IN+ gain** 选项，以满足窗口需要的高侧阈值。
2. 重复步骤 1 配置比较器 ACMP3L，以满足窗口需要的低侧阈值。
3. 将 ACMP3L 的 **IN+ source** 配置为 **ACMP2L IN+ source**，使得比较的是同一个输入源。
4. 添加 LUT 逻辑使得当窗口低侧阈值的比较器输出低或者高侧阈值的比较器输出高时，触发中断。

应用：过温保护

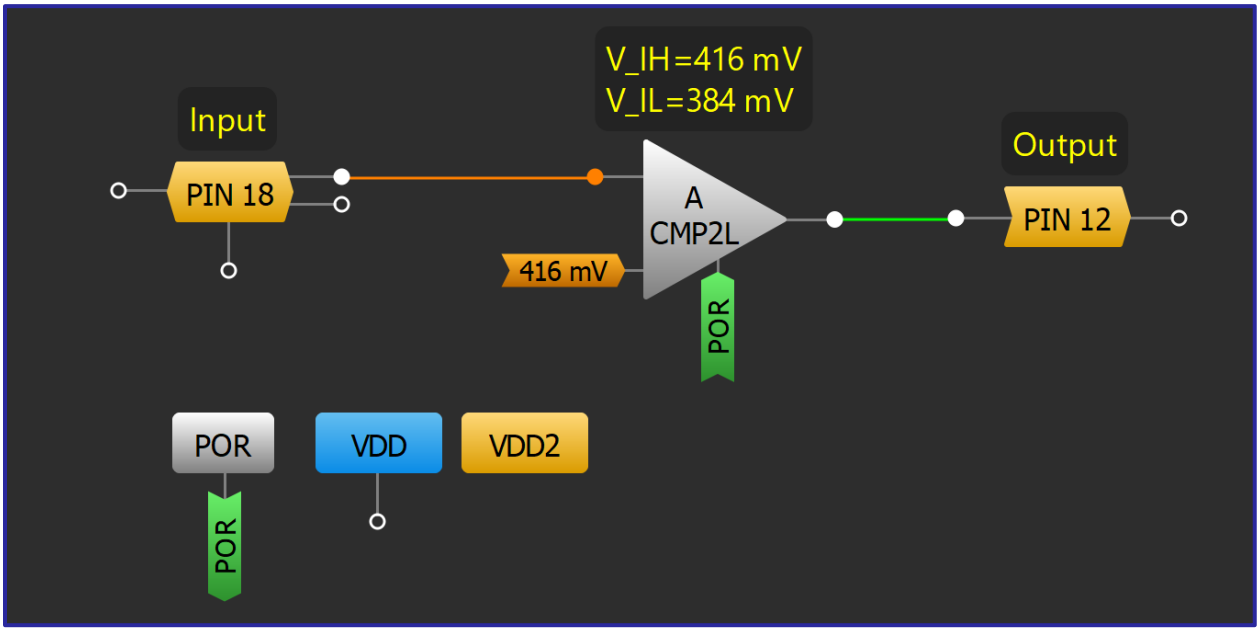
过温保护电路广泛应用于高温状态提醒系统，当内部温度超过安全阈值时，该电路可以防止系统过热。



所需元器件

- 任意带有 ACMP 的 GreenPAK IC(CMIC)
- 1 个电阻
- 1 个 NTC 型热敏电阻

GreenPAK 设计图

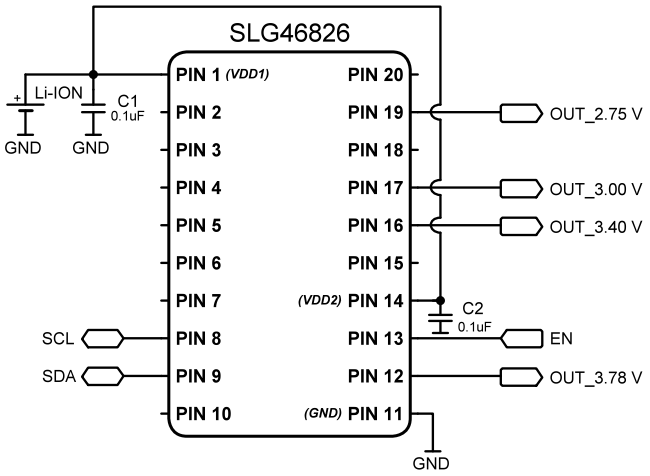


设计步骤

1. 配置 **ACMP2L** 的 **IN+ source** 为 **Pin18**， **IN- source** 为所需的安全阈值。
2. 将一个电阻的一端连到 VDD 另一端连到 **Pin18**。
3. 将 NTC 电阻一端连接到 **Pin18** 另一端连接到 GND。

应用：充电指示器

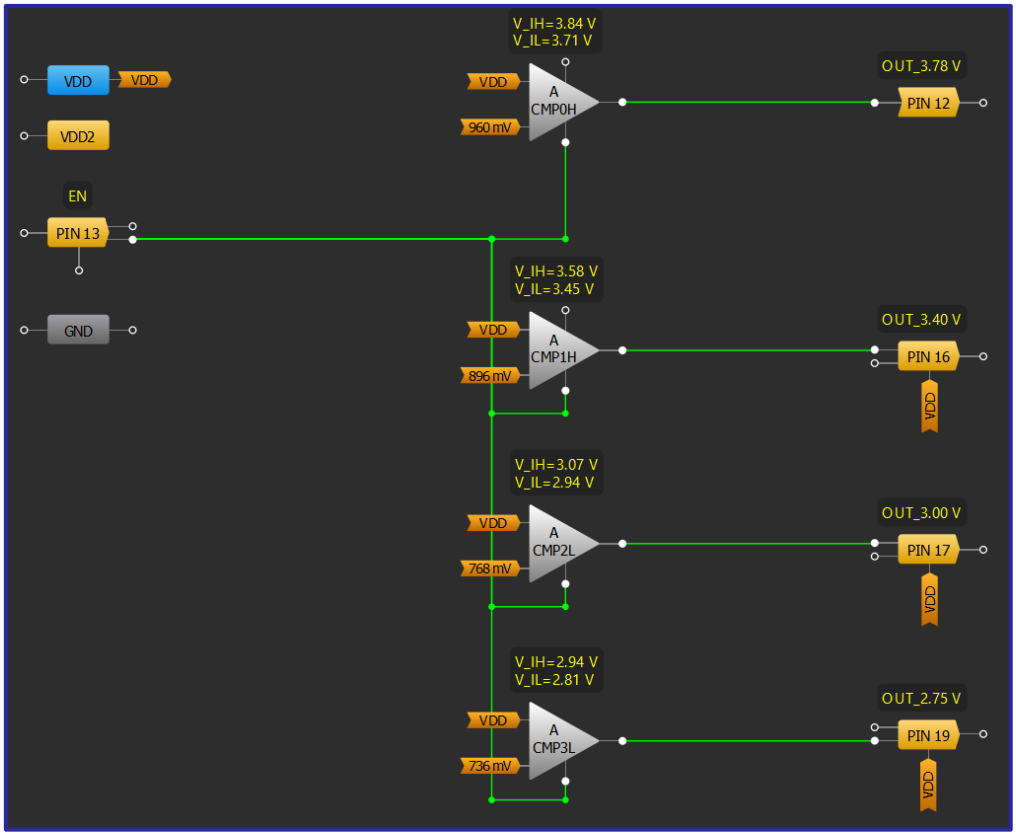
充电指示器用于指示电池供电设备的充电状态，该设计适用于锂电池供电设备。



所需元器件

- 任意带有 ACMP 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

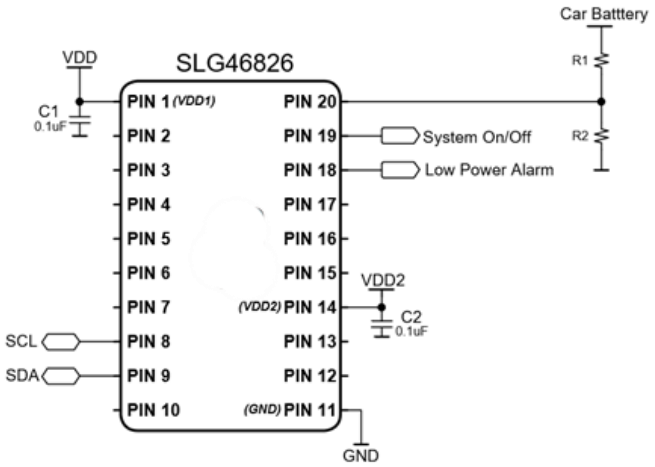


设计步骤

1. 将 ENABLE 信号连接到所有 ACMP 的 **PWR UP** 脚。
2. 将所有 ACMP 的 **IN+ source** 设置为 VDD/PIN20，并将每个 ACMP 的 **IN- source** 设置为所需的阈值电压。

应用：信息娱乐系统低压指示器

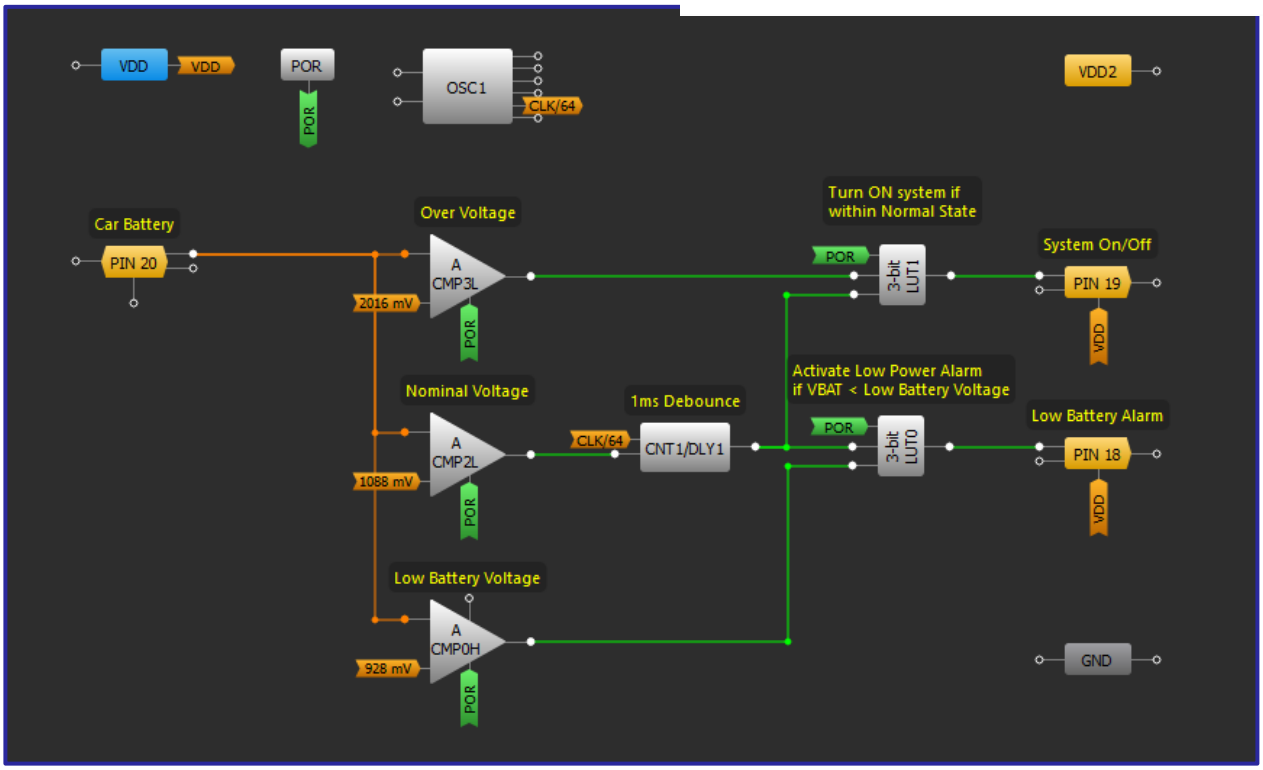
电压指示器用于电池供电设备，用来指示充电状态。该设备可以监测汽车电池的电压水平，并根据需要调整信息娱乐活动以节省电力。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 两个分压电阻

GreenPAK 设计图

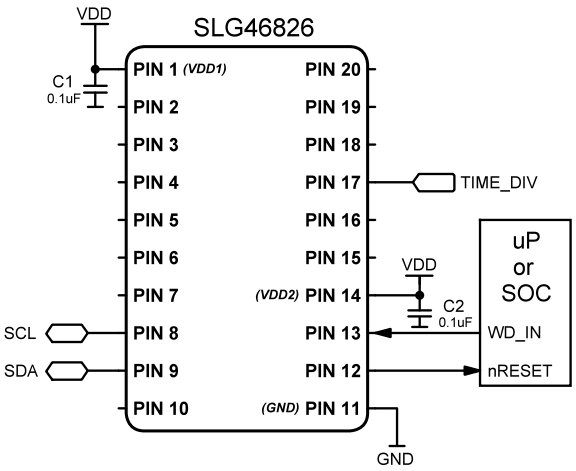


设计步骤

1. 将 **ACMP0H IN+**端配置为 **PIN 20**，其它 ACMPs 的 **ACMP0H IN+**端配置为 **ACMP0H IN+ source**。
2. 为 **PIN 20** 添加一个电压分压器以处理来自汽车电池的高电压。
3. 将 **IN- source** 配置为所需的电压阈值。
4. 配置逻辑单元以确定输出的电压窗口。
5. 在 **ACMP2L** 和 **3-bit LUT0** 之间添加去抖延迟。

应用：看门狗定时器

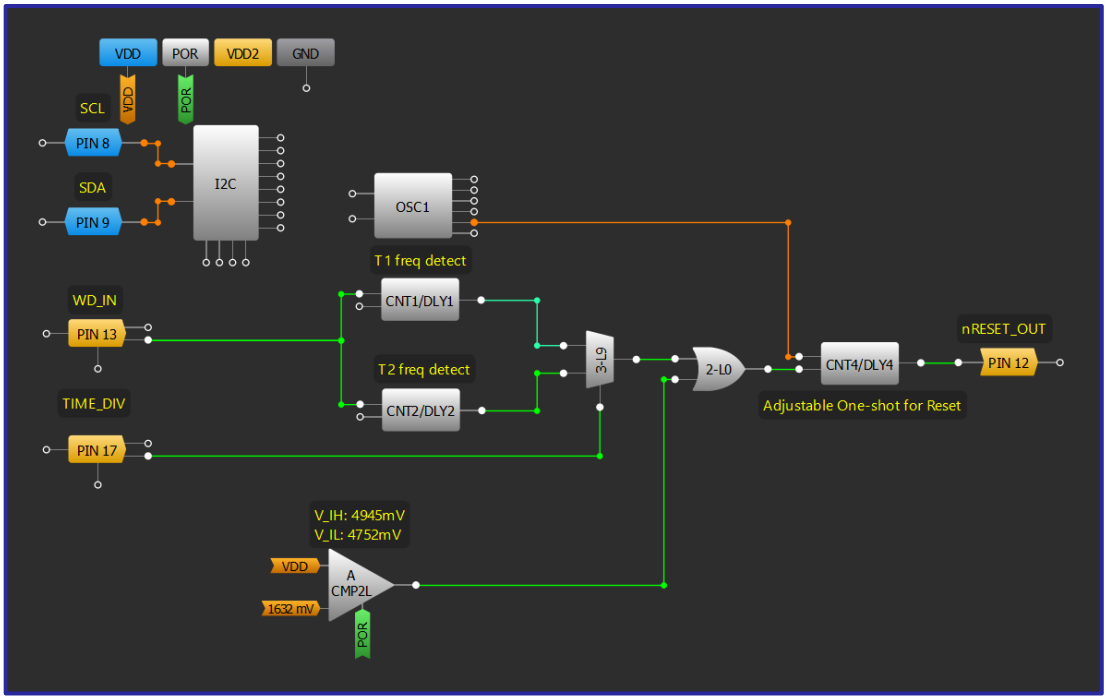
如果微控制器或者微处理器没有周期性发送喂狗信号，看门狗定时器将产生一个系统复位信号，看门狗芯片一般还附带监测系统电压的功能。



所需元器件

- 任意带有 ACMP 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

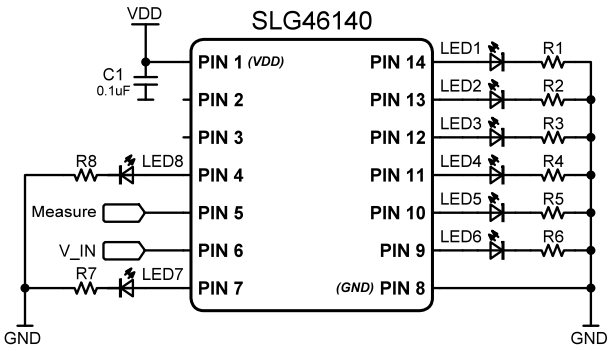


设计步骤

1. 用 ACMP 设置看门狗的欠压阈值。
2. 将一个 CNT 配置为 **Frequency detects** 模式。
3. 设置数字逻辑，以组合欠压和看门狗超时的有效信号。
4. 添加一个 One-shot 模式的 CNT 以输出所需脉冲宽度的复位信号，可以反转为低电平有效的输出。

应用：电压检测

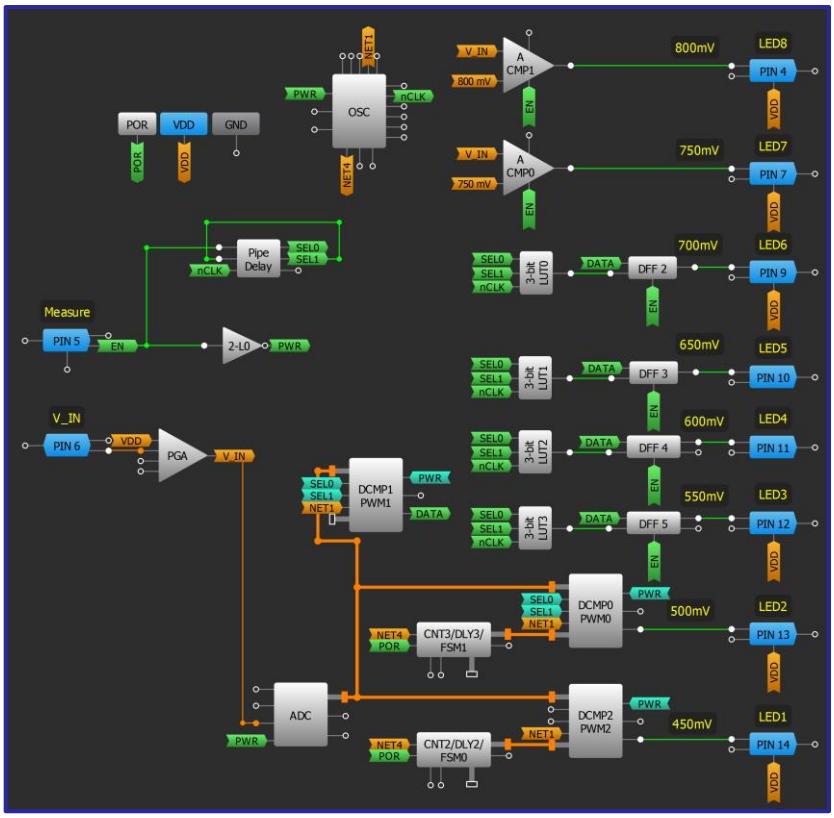
在某些设计中，系统需要区分开/监控电压多种阈值等级，而不仅仅是极少数几个不同电压等级。本设计阐述了使用带有 ACMP、DCMP 和 ADC 的 GreenPAK 进行电压幅值检测。



所需元器件

- 任意带有 ACMP, DCMP 和 1 个 ADC 的 GreenPAK IC(CMIC)
- 8 个 LED 和电阻

GreenPAK 设计图

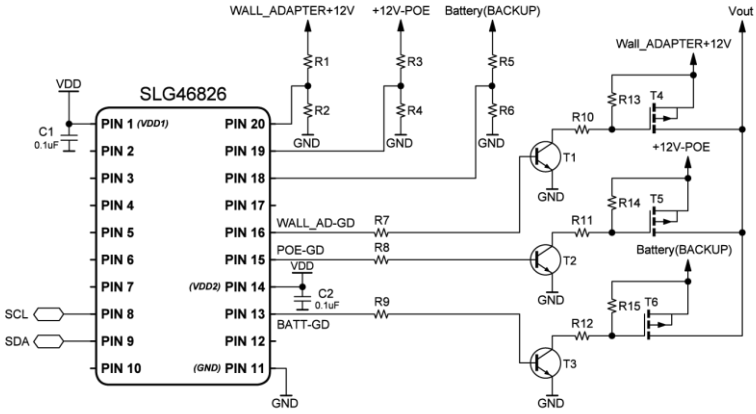


设计步骤

1. 启用 ADC，DCMP 和 ACMP 模块。
2. 使用 技巧：PWM 模式下的 DCMP PWM 模块应用 配置 DCMP。
3. 将每个 ACMP 和 DCMP 的 **IN- source** 设置为所需的阈值电压。
4. 添加 LUT 和 DFF 逻辑单元，用以选择和写入来自于 **DCMP1** 中电压幅值，从而区分开不同幅值的电压等级，并将对应等级用 GPO 输出。

应用：电源后备管理

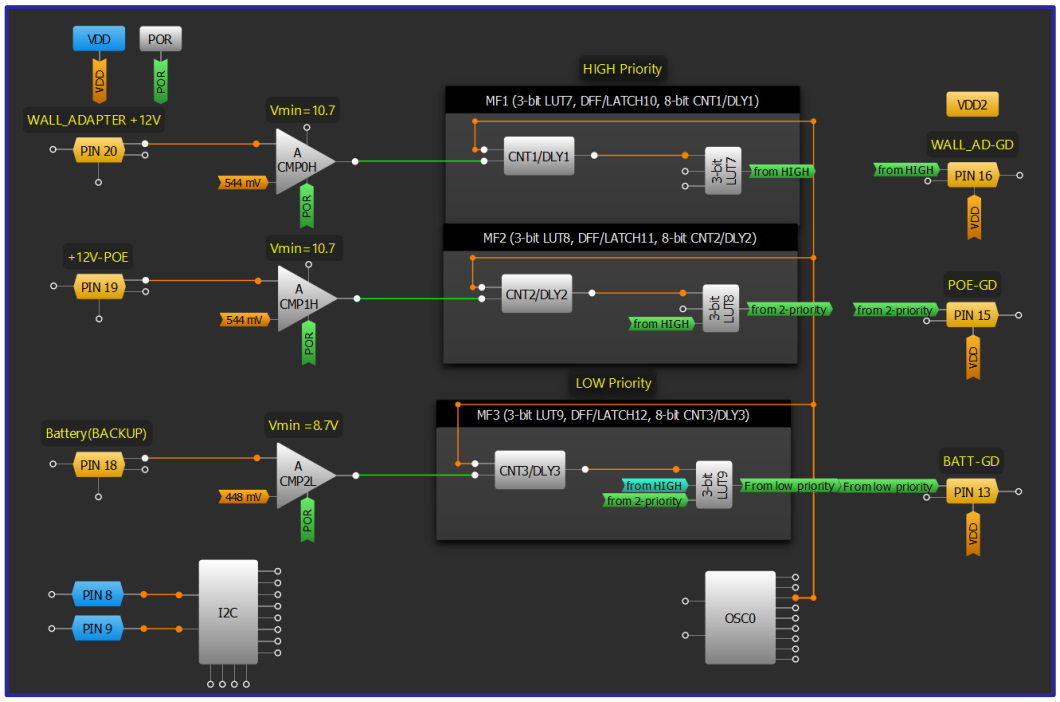
电源后备管理用于单一电源输入的设备使用多路电源供电时，需要为设备提供不间断供电的系统，防止因为某一路电源异常而造成设备掉电的情况。



所需元器件

- 任意带有 3 个 ACMP 的 GreenPAK IC(CMIC)
- 外接电阻分压器以将输入电压调整至 ACMP 的工作电压范围

GreenPAK 设计图

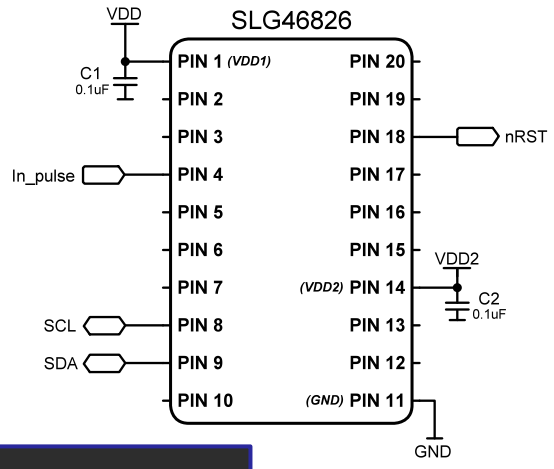


设计步骤

1. 使用 3 个 ACMP 以检测不同的电源输入信号。
2. 配置一个 CNT/DLY 模块为 delay 模式以建立去抖滤波。
3. 添加逻辑单元以建立不同电源系统的切换优先顺序。

应用：N 脉冲看门狗

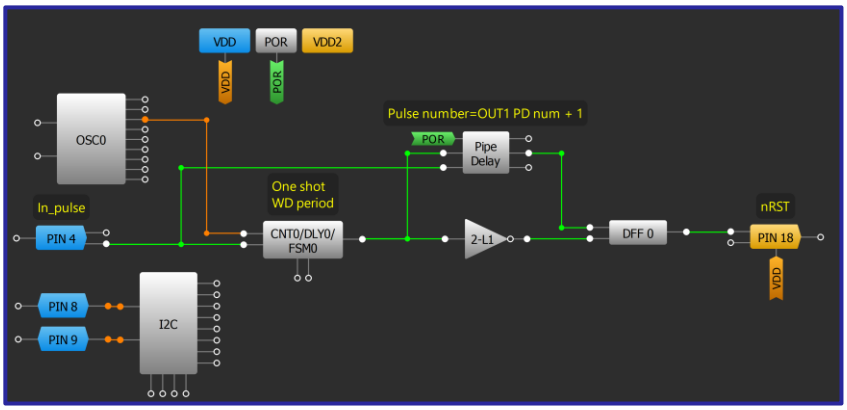
看门狗定时器用于在微控制器或微处理器无法定期发送脉冲时自动产生系统复位信号。此应用可监测在看门狗周期内进入 GreenPAK 的脉冲数。如果脉冲数量小于预定义的脉冲数，将触发系统复位。

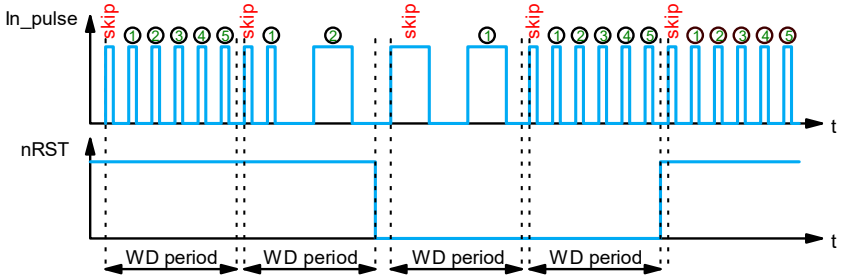


所需元器件

- 任意 GreenPAK IC(CMIC)

GreenPAK 设计图





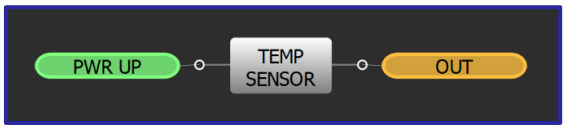
设计步骤

1. 将 **CNT0/DLY0/FSM0** 配置为在所需的看门狗周期内的一个单脉冲发生器。
2. 在 Pipe Delay 中定义脉冲数量（注意：*Pulse number = OUT1 PD num + 1*）。
3. 将单脉冲发生器(**CNT0/DLY0/FSM0**)的输出反相并连接到 **DFF0** 的 CLK 输入端。
4. 将 Pipe Delay 的 **nOUT1** 连接到 **DFF0** 的 **D** 输入端。

技巧：温度传感器模块的应用

此技巧适用于所有带有温度传感器宏单元的 GreenPAK 芯片。

某些 IC 包含一个模拟**温度传感器 (TS)**，其输出电压与摄氏温度成线性比例。**温度传感器**的额定工作温度范围为 -40°C 至 85°C。整个温度范围内的误差不超过±0.85%。**温度传感器**的输出可以直接连接到**模拟输出**或模拟比较器的正输入端。**温度传感器**可以有两个输出电压范围和一个**上电(使能)输入**端子(PWR UP)。**上电(使能)输入**可以选择使用(芯片内部)矩阵输入或使用寄存器激活(使能)。温度传感器也可以通过 I²C 激活并更改温度范围。在恒定温度下温度传感器输出电压随 VDD 变化的变化非常小（例如，在 SLG46826 中，输出电压误差在所有温度下均小于 ±0.08%）。



温度传感器宏单元

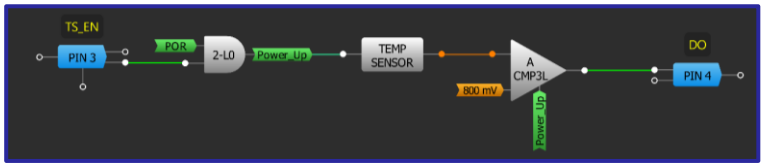
温度传感器的输出电压可以通过以下公式计算：

$$V_{ts} = K \times T + V0$$

其中：
Vts - 温度传感器输出电压
K - 系数
T - 摄氏温度
V0 - 0°C 时的输出电压

温度传感器连接到模拟输出

温度比例电压信号可以作为到**模拟输出(16 脚)**。**Power down source** 配置应设置为“From register”。
温度传感器输出的信号可以与模拟比较器模块中的参考电压进行比较(见下图)，在离散输出处产生双态信号。为了降低功耗，启用了 **TS_EN**。**TS_EN** 使能温度传感器，打开 **ACMP3L**。**Power down source** 应设置为“From matrix”。



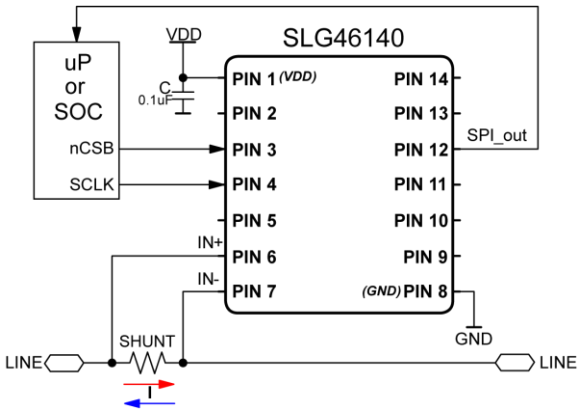
温度比较器

带有温度传感器宏单元的 GreenPAK 芯片可以实现：

- 测量 PCB 元件的温度
- 测量 FET 或晶体管外壳温度
- 为 SoC 或在闭环应用中创建报警信号
- 最小化 adc, dac, OpAmps 和其他温度相关器件的误差

应用：通过外部采样电阻进行电流检测

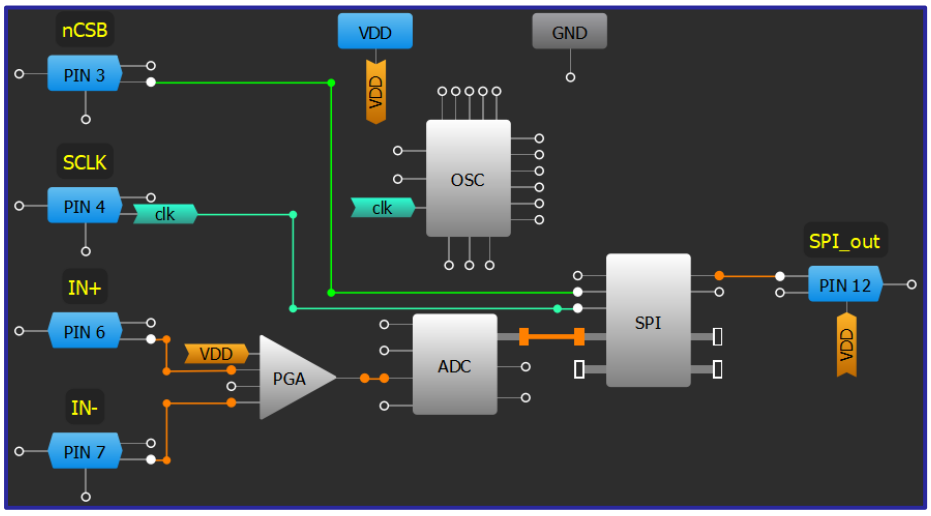
GreenPAK 可应用于通过检测电阻的电压来检测通过设备的电流。此应用输出一个串行代码来表示它的采样值。



所需元器件

- 任意带 PGA，ADC 和 SPI 的 GreenPAK IC
- 1 个电阻

GreenPAK 设计图

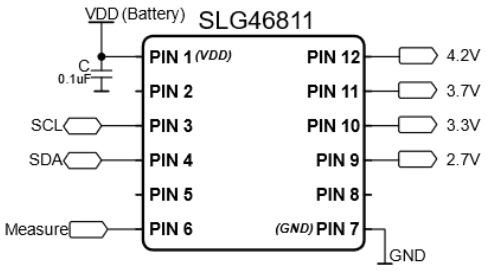


设计步骤

1. 通过移除 **PWR DOWN** 输入端子上的 VDD 来启动 ADC。
2. 将 PGA 配置为“Differential”差分模式。 它将自动连接 ADC 相应输入端子到 **PIN6** 和 **PIN7**。
3. 将 SPI 设置为“P2S”模式，将 **PAR input data source** 设置为“ADC”。

应用：使用一个复用 ACMP 监控一个模拟量的四个电平

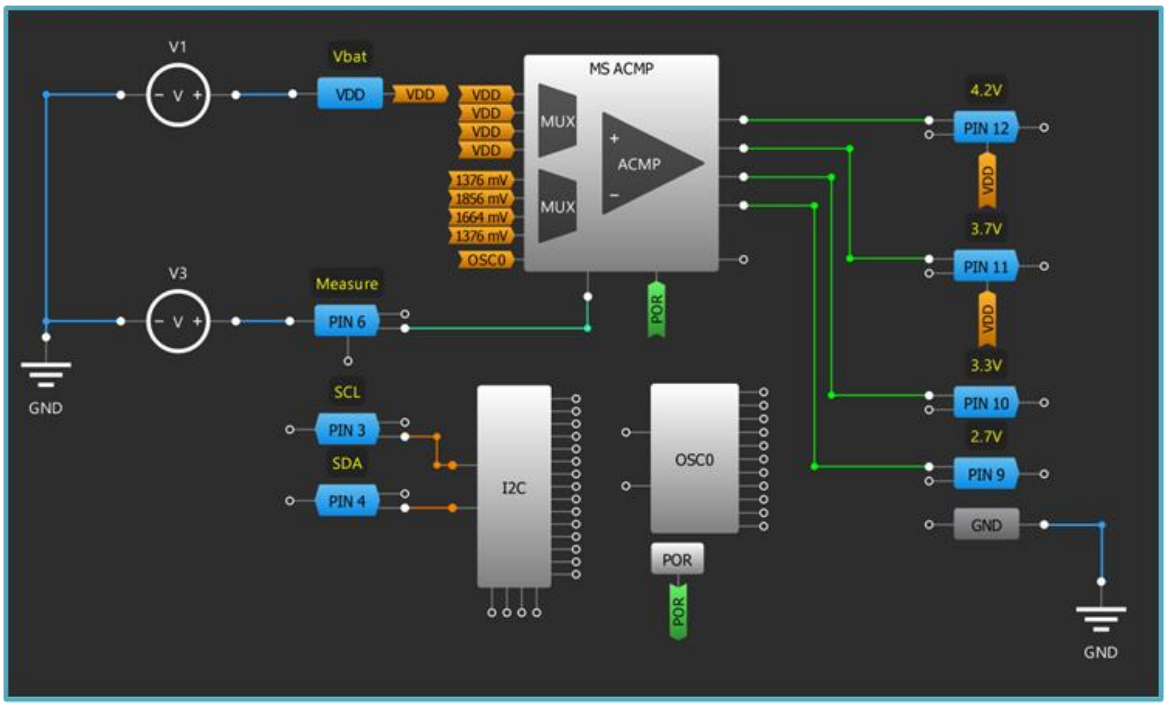
用 GreenPAK 监测一个模拟信号的四个电平在各种应用中都被广泛使用。例如，它可以用于电池电源管理、液位控制、温度/亮度/距离/压力/湿度检测等。



所需元器件

- SLG46811V 或带有相应数量 ACMPs 的任一 GPAK

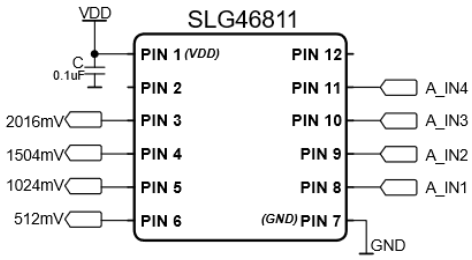
GreenPAK 设计图



设计步骤

1. 配置 MS ACMP 的 ACMP Mode 为 Multi-channel 模式，Number of Channels 选择 4。
2. 选择 Enable mode 为 Rising Edge Activation，使能 MS ACMP。
3. 为 Channel 0 – Channel 3 调整适当 IN- source 参数。
4. 配置 PIN6 为数字输入(digital input)，连接到 MS ACMP 的 Enable 引脚。每当 PIN6 脚出现上升沿时，MS ACMP 都将 VDD 电压与 4 个阈值相比较，其结果输出到对应端口 PIN9-PIN12。
5. **I²C** 可以重写 ACMP 的阈值。

应用：用一个 MS ACMP 监控四个独立模拟信号

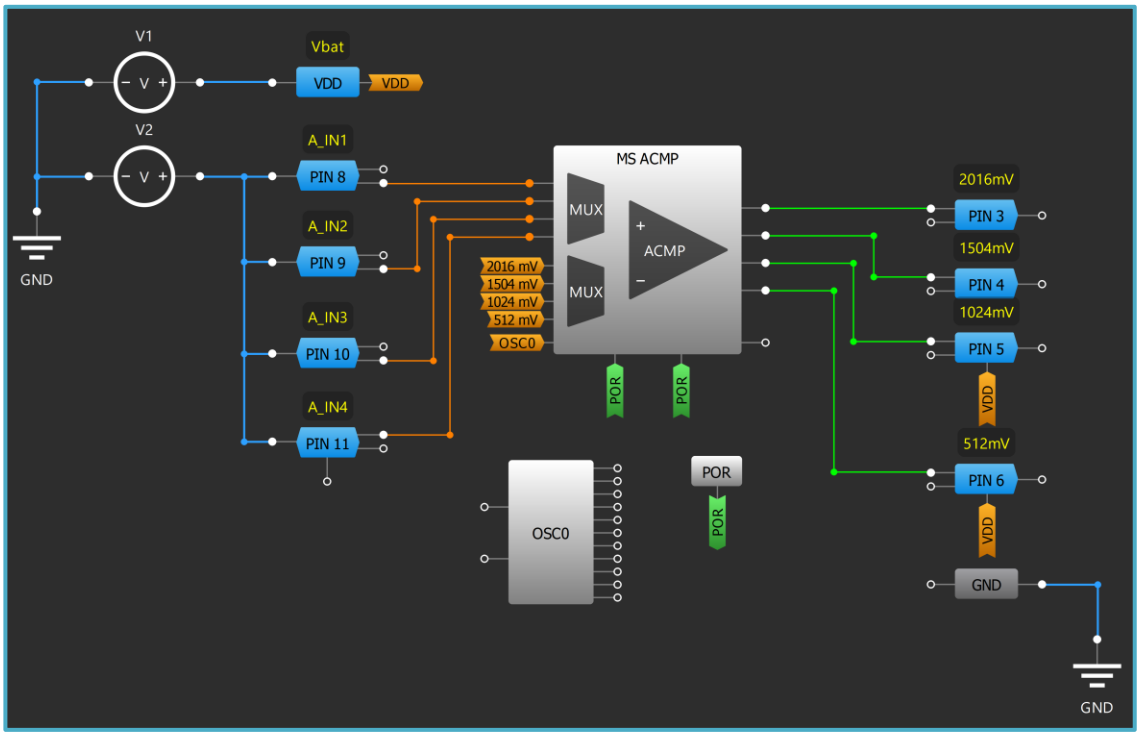


使用 SLG46811V 中的 MS ACMP 宏单元或其他 GreenPAK 芯片中 4 个单独的 ACMP 来监控四个不同的模拟信号。

所需元器件

- SLG46811V 或带有相应数量 ACMPs 的任一 GPAK

GreenPAK 设计图



设计步骤

1. 配置 MS ACMP 为 ACMP Mode 为 Multi-channel 模式，Number of Channels 选择 4。
2. 选择 Enable mode 为 High-Level Activation，并连接到 POR (MS ACMP 将连续采样)使能 MS ACMP。
3. 为 Channel 0 – Channel 3 调整适当 IN- source 参数。将 PIN8 - PIN11 的电压与参考电压进行比较，并相应地将比较结果输出到 PIN3-PIN6。
4. **I²C** 可以重写 ACMP 的阈值。

第 5 章: 通讯协议

本章介绍了器件之间的通讯应用。涉及到以下应用和技术：I2C、串行、并行通讯协议。

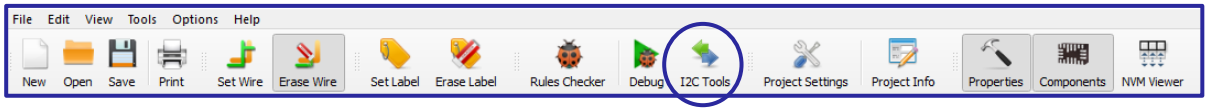
本节中提供的许多技巧和应用都依赖于GreenPAK 的I²C 功能，要了解GreenPAK 中I²C 的详细信息，请查阅芯片的数据手册。

技巧： 利用 I²C 更改设计

该技巧可以应用于任何包含 I²C 接口的GreenPAK 中。

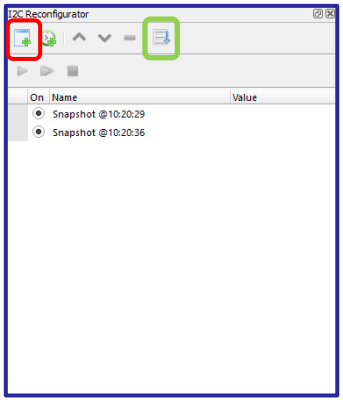
如果该 GreenPAK 带有 I²C 接口，那么在芯片烧录之后，还可以通过 I²C 接口实时编辑芯片的功能，但是如果需要在掉电后保存设计更改，需要设备支持 MTP 和 ISP[In-System Programming]，该技巧提供了一种快速了解更改设计所需 I²C 命令的方法。

1. 利用 GreenPAK Designer 完成你的初始设计，该设计将会在上电时装载并运行。
2. GreenPAK Designer 中, 选择 **I²C Tools** 按钮以打开 **I²C Reconfigurator** 窗口(新版软件，需要先点开 debug)。



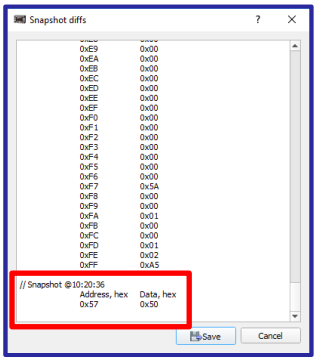
I²C 工具按钮

3. 在 **I²C Reconfigurator** 窗口点击 **snapshot** 快照按钮 (右图红框), 或者按下快捷键 SHIFT+A，这将产生一个对应当前设计的 I²C 命令行表快照。
4. 将您的设计更改为下一个配置。
5. 利用步骤 3 的方法为第二个设计产生一个快照。
6. 点击 **Snapshot Diffs** 快照差异按钮(右图绿框)，这将显示创建此设计需要的 I²C 命令。
7. 向下滚动 **Snapshot diffs** 命令行，找到第二个快照，这将仅显示第一个设计快照和第二个设计快照之间的差异。



I2C Reconfigurator

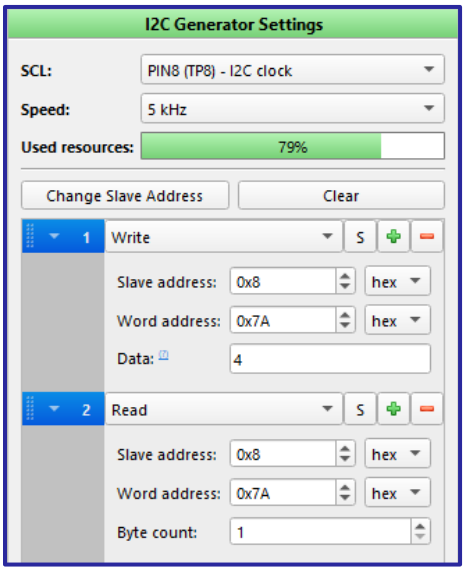
8. 这些差异如右图红框所示，显示了改变 ACMP 的阈值(或者其他可能的更改)需要通过 I²C 接口发送的地址和数据信息，以十六进制显示。



Snapshot Diffs

技巧：创建一个 I²C 命令

此技巧可以应用于任何包含 I²C 宏单元的 GreenPAK 芯片。



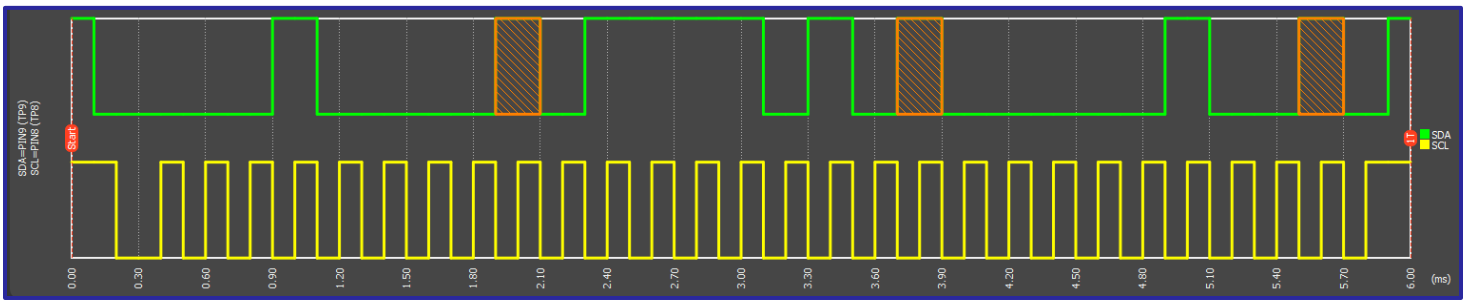
利用 I²C generator，通过逻辑信号生成器，设计者可以生成 I²C 信号。它由 SDA 和 SCL 两组逻辑信号构成。 设计者可以通过将预先定义的 I²C 信号进行组合而产生需要的信号波形，并选择 SCL 的频率：

对于 GreenPAK Advanced Development 平台，可选频率为 1k, 2.5k 和 5k Hz ；

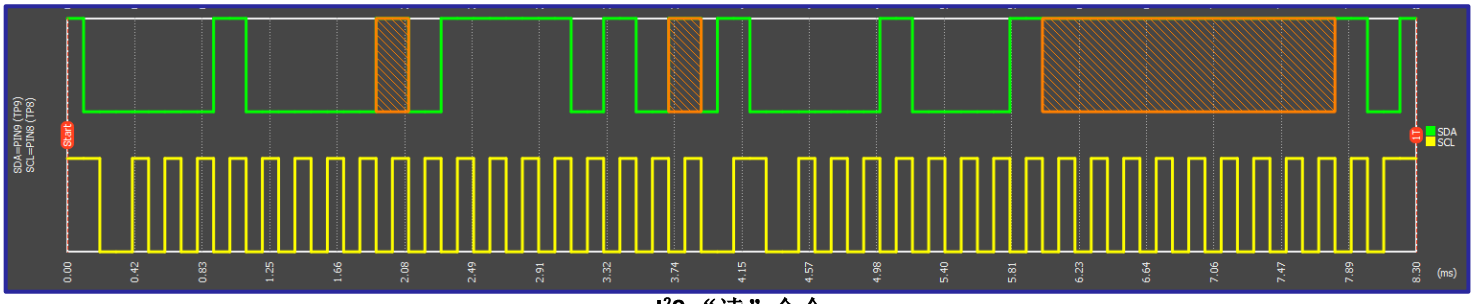
对于 GreenPAK Pro Development 平台，可选频率为 1k, 2.5k, 5k, 10k, 20k, 50k, 100k, 200k, 400k, 1000 kHz。

使用 **I²C generator** 通过以下步骤生成一个 I²C 信号：

1. 选择 Debug 按钮。
2. 选择“ **I²C generator**”仿真信号并添加到 I²C 模块的 **SDA** 引脚的外部输入端。
3. 单击 **EDIT** 转到信号配置向导。
4. 选择 PIN8 作为 SCL 输入 并设置时钟频率。
5. 选择 Read 或 Write 复合命令。
6. 展开复合命令并设定 Slave address（设备地址）和 Word address （字地址）。 对于“**读**”命令，设置字节数。 对于“**写**”命令，设置需要写入的数据。



I²C “写”命令

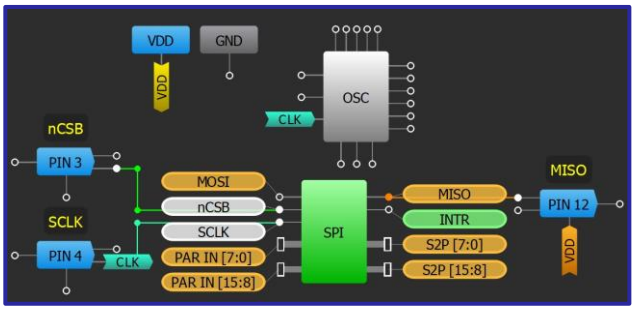


I²C “读”命令

技巧：串行-并行接口 (SPI) 模块

*此技巧介绍了**串行转并行接口 (SPI)** 模块，适用于 SLG46140, SLG46620 和 SLG46621。*

SPI 模块是一种特殊的宏单元，可用于 GreenPAK 和 SOC （系统级芯片）之间的通讯。 它可以将串行数据转换为并行数据，也可以将并行数据转换为串行数据。其输入是标准的 SPI I/O 接口 (**MOSI**, **MISO**, **nCSB**, **SCLK**, and **INTR**)。**nCSB** 是低电平有效片选信号。**SCLK** 是 SPI 宏单元的串行时钟信号。



SPI 宏单元

SPI 可以传输数据到如下模块：

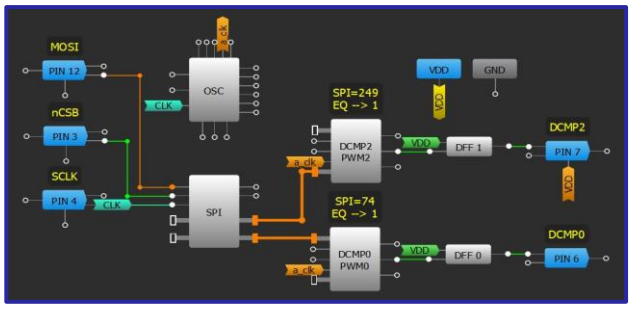
- FSM
- DCMP
- DAC (through DCMP)

同样地，SPI 可以传输来自以下模块的数据：

- ADC
- FSM

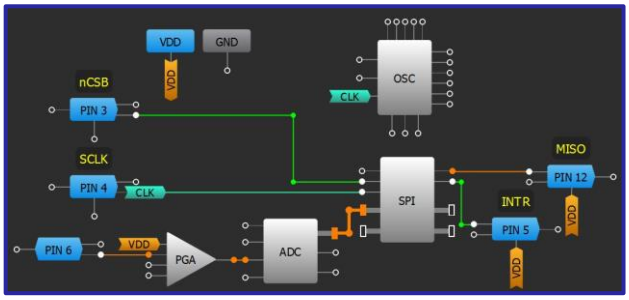
SPI 可以和其他宏单元组合实现如下功能：

- 脉宽调制
- 模数比较
- 数模比较
- 结合 DCMP，比较两种结果
- SDIO 和 LCD



SPI 宏单元将串行数据转换为并行数据

SPI 可工作于 8 位或 16 位模式。需要注意的是 SPI 宏单不能在同一个程序文件中发送和接收串行数据。必须设置为 “S2P” 或 “P2S” 模式。



SPI 宏单元将并行数据转换为串行数据

在 P2S 模式下，每次数据传输完成后，**INTR** 脚将输出一个周期高电平。 此外，GreenPAK 芯片中的 SPI 模块符合通用标准。 其时钟频率最高可达 2 MHz。**CPOL** 决定时钟脉冲的有效脉冲方式（正脉冲、负脉冲），**CPHA** 定义数据传输和串行时钟间的相位关系。 当 CPHA = 0， 数据仅允许从串行到并行传输，不允许从并行到串行传输。 当 CPHA = 1， 数据既可以从串行到并行传输，也可以从并行到串行传输。

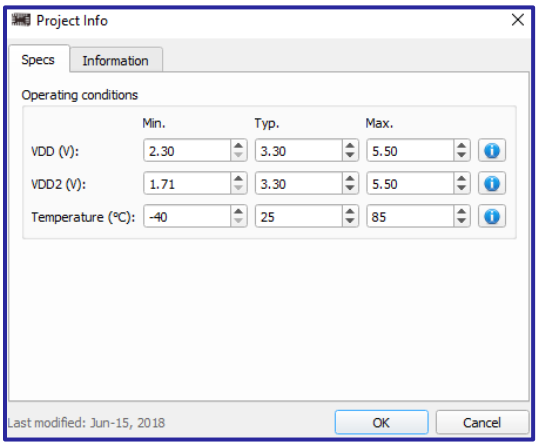
技巧：电平转换

该技巧适用于任何具有双电压轨的GreenPAK，例如SLG46826V。

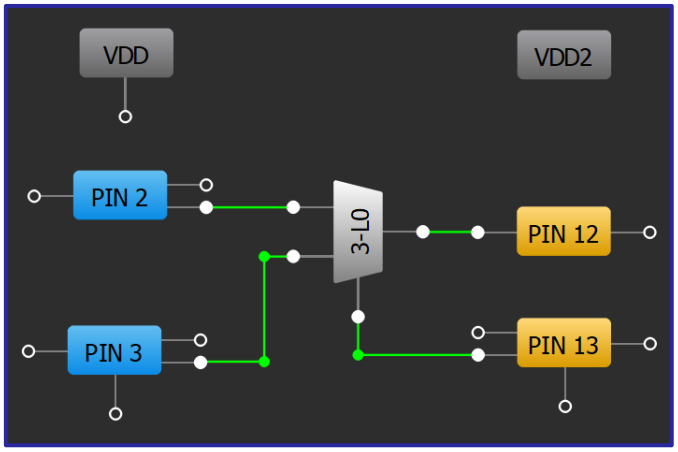
通常在系统级设计中，需要组合来自两个不同电平的信 号，例如模拟电压轨工作在 5V ,而数字电压轨工作在 3.3V。

许多 GreenPAK 通过使用双电压轨来解决这个问题，来自不同电平的信号都可以输入到 GreenPAK，进行处理，然后从任意电压轨输出。

当使用双电压轨的器件来开始进行一个设计的时候，需要如右图所示的那样，分别输入 2 个电压轨的电压范围， 两个电压轨的工作范围随不同器件而有所不同，但是 VDD 轨总是电压较高的轨，而不是 VDD2 轨。



双电压轨项目信息



双电压轨逻辑示例

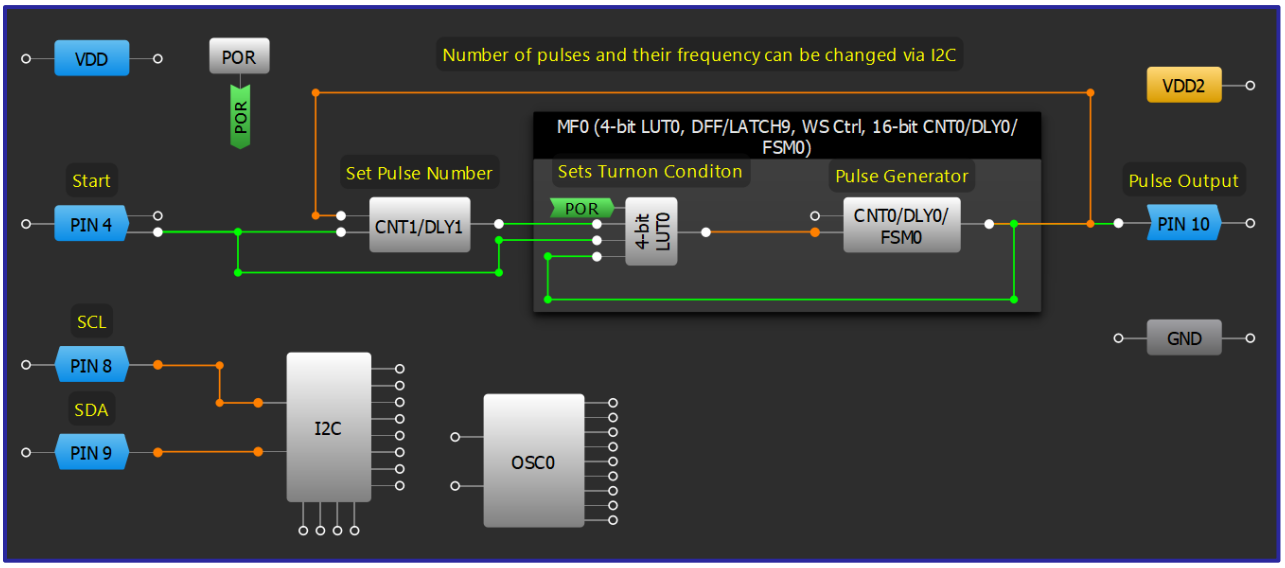
如左边的图所示，在 GreenPAK Designer 中处在不同电压轨的 GPIO 使用不同的颜色来表示， 处在 VDD 轨的 GPIO 用蓝色显示，处在 VDD2 轨的 GPIO 用琥珀色来显示，不同电平的信号在进入 GreenPAK 矩阵后都将表现一致。

技巧：发送预设数量脉冲

此技巧适用于所有GreenPAK，但是含有Multi-Function 组件的GreenPAK 可以降低设计复杂度。

在许多通信协议中都需要由一个 IC 发送或者接收一定位数的比特数据，这通常意味着 GreenPAK 需要实时跟踪发送或者接收到的脉冲个数，例如利用一个移位寄存器来接收数据的时候，需要监测接收到的位数， 而不会错误地偏移和失控地不停移位，以确保寄存器中存放的是正确的数据。

在 GreenPAK 中设置预定数量的脉冲有很多办法。此技巧描述了一种有效的并且方便扩展位数的方法。此方法还通过每次传输完成后都重置时钟偏差来限制 GreenPAK 和其他芯片之间累积的时钟偏差。下图显示了此方法类似流水线结构的 GreenPAK 设计图。它包括两级电路：一个脉冲计数单元和一个脉冲发生器。



预设脉冲发生器设计图

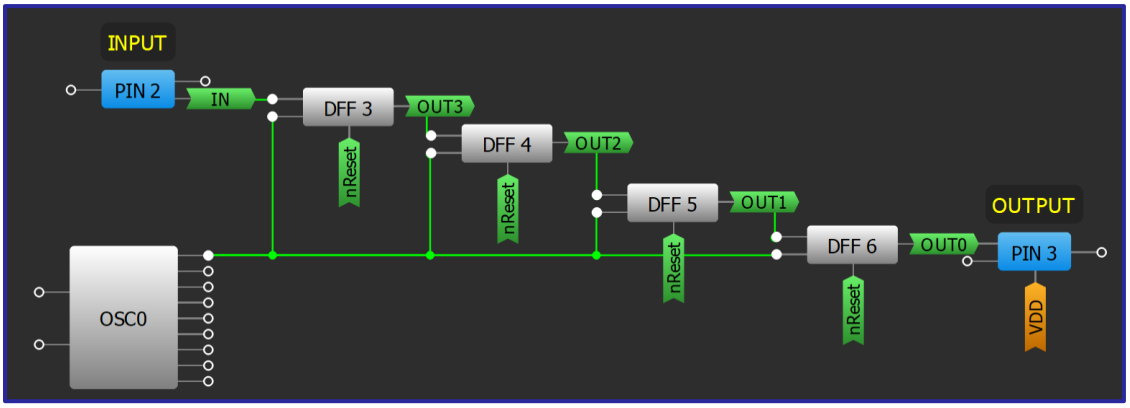
脉冲计数单元由一个单次脉冲模块组成，该模块由脉冲发生器的输出脉冲提供时钟源。在 **PIN 4**（Start）的上升沿，**CNT1/DLY1** 输出将设置为高电平，直到输入时钟达到 **Counter data** 设置的脉冲数量。在达到设定的脉冲数后，它将返回 **LOW**。

脉冲发生器采用 **MF0**。在 **MF0** 中，**CNT0/DLY0** 是一个带有反相输出的双沿延迟。它的延迟时间即是设置脉冲发生器的周期。其输出反馈到 4 位 **LUT0**，该 **LUT0** 配置为仅在 **CNT/DLY1** 的输出为高电平时反转来自 **CNT0/DLY0** 的信号。(**CNT0/DLY0**) 单次脉冲完成后，脉冲发生器将停止发送脉冲。

技巧：移位寄存器

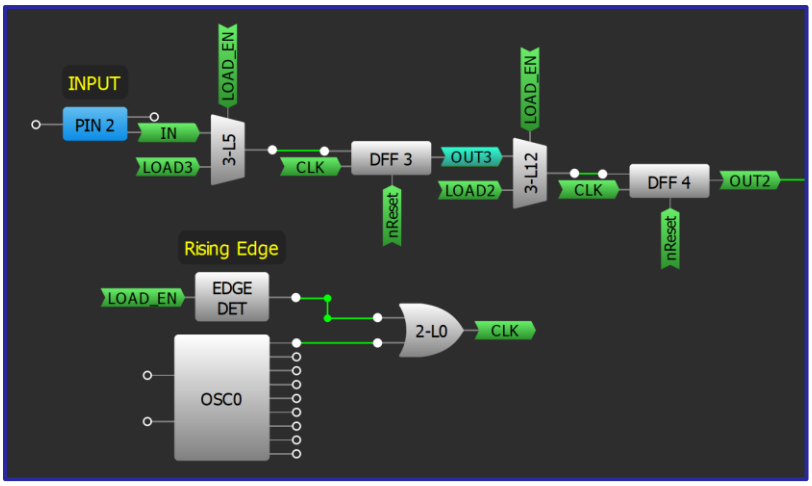
此技巧适用于任意 GreenPAK IC (CMIC)，能够配置成多少位数的移位寄存器取决于 GreenPAK 中可用的组件。

移位寄存器是串行器和解串器关键的组成部分，移位寄存器就是一串触发器的集合，并且每个触发器的输出都可以单独访问，所有触发器都共享一个时钟，在这个时钟的上升沿，寄存器按顺序将数据移位到下一个触发器，下图显示了一个基本的 4 位大小的移位寄存器，并且可以通过一个共享的复位信号进行全局复位。



基本移位寄存器

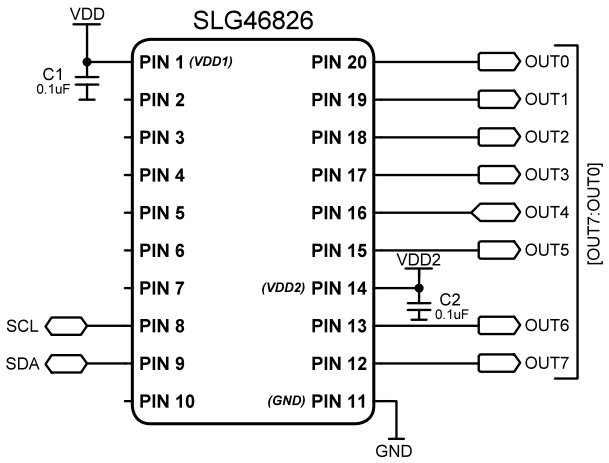
通常，移位寄存器需要能够装载初始值，所以需要在每个 DFF 之前加一个 MUX，当预载数据已经准备好时，切换 MUX 的输入通道（GreenPAK Designer 中的 **S**），然后触发 DFF 的时钟，完成装载，下图显示了如何从上述设计中的基本移位移位寄存器添加 MUX 形成可以装载的移位寄存器，**Load_EN** 被用来切换 MUX 的输入外，还用来触发 **DFF3** 和 **DFF4** 以完成装载。



可以装载的移位寄存器

应用：通过 I²C 接口进行 IO 扩展

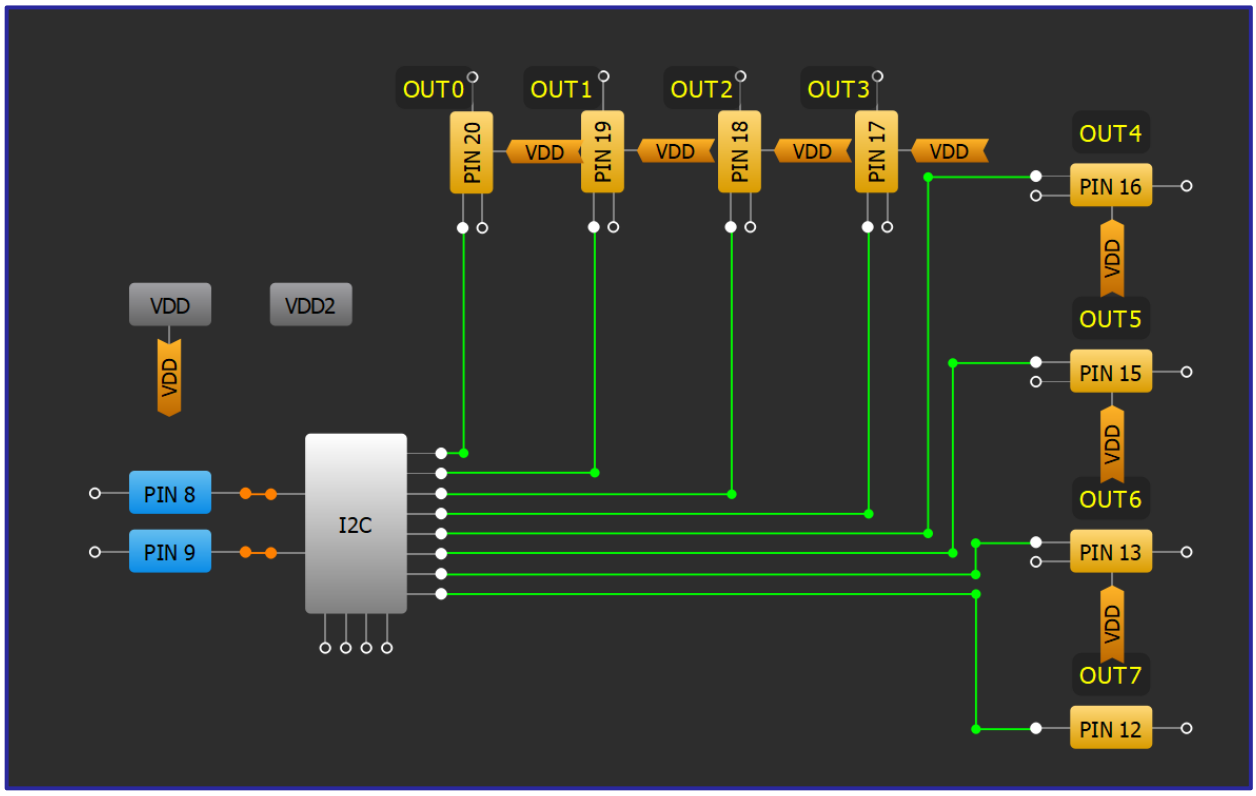
IO 扩展器用于使用较少的线控制更多的 IO，因为 I²C 可以通过使用不同的地址将扩展器连接在同一条总线上以扩展更多的 IO，所以是 IO 扩展器中一种常见的输入接口。



所需元器件

- 任意带 I²C 接口的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

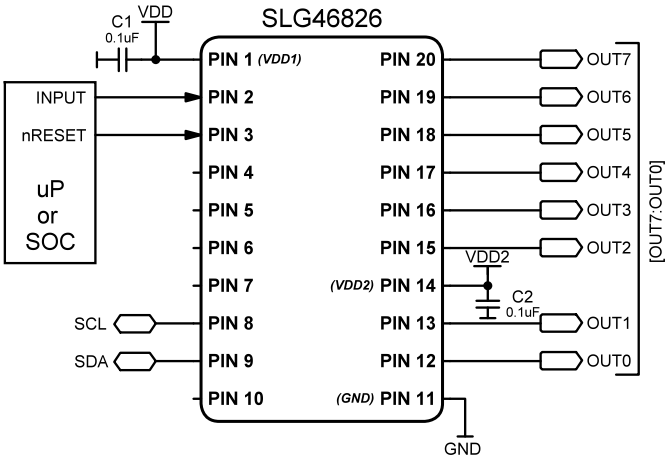


设计步骤

1. 将 GPIO 配置为 **Digital output**。
2. 将配置好的 GPIO 连接到 **I²C virtual inputs**。
3. 通过访问 **I²C Virtual inputs** 对应的寄存器[寄存器地址参见特定 GreenPAK 数据手册的 Connection Matrix Virtual Inputs 部分]，每个 I²C Virtual Input 都可以单独更改或者同时更改。

应用：串口转并口(外部时钟)

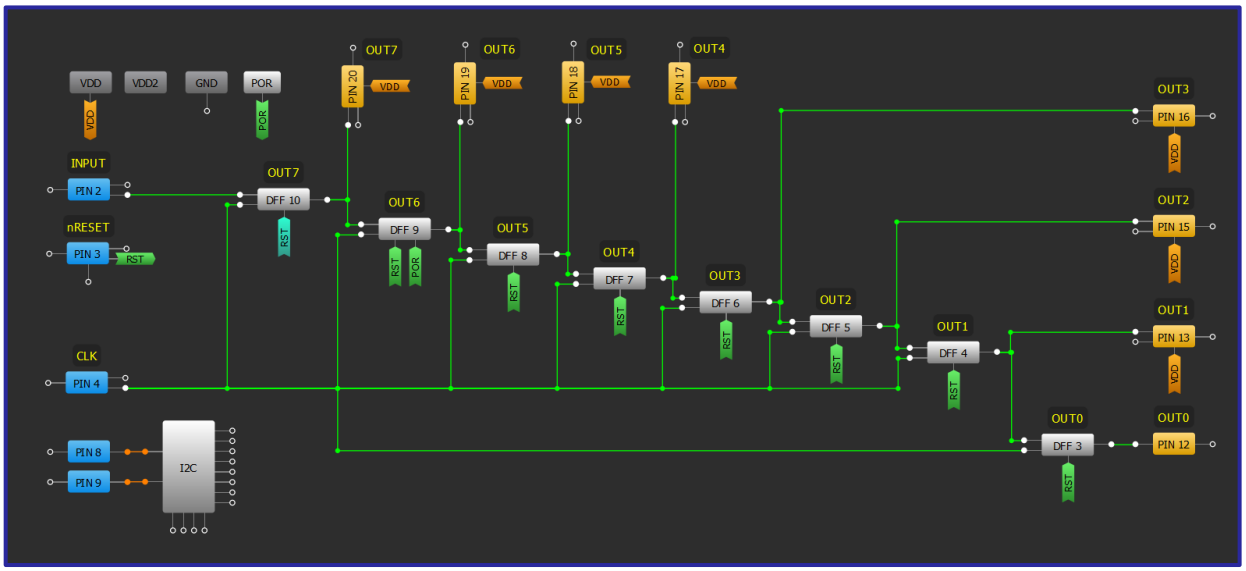
解串器[串口转并口]是将单线发送的串行多位数据由接收器解码为并行数据，通常还需要添加一条时钟线以避免不正确的数据移位。



所需元器件

- 任意 GreenPAK IC (CMIC)
- 1 颗可以输出时钟信号的芯片

GreenPAK 设计图

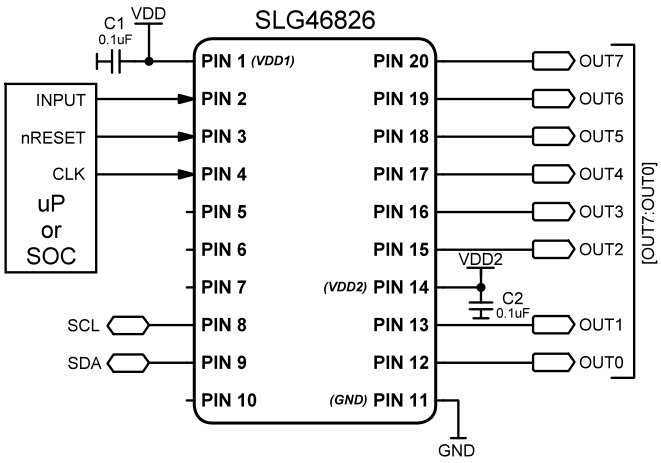


设计步骤

1. 使用 技巧：移位寄存器 将移位寄存器配置成 GreenPAK 设计图那样。
2. 为时钟信号配置一个 **Digtal input** 引脚，并将其连接到移位寄存器 **CK**[组成寄存器 DFF 的 **CK**]。
3. 为复位信号配置一个 **Digital input** 引脚，并将其连接到移位寄存器 **nRESET**[组成寄存器 DFF 的 **nRESET**] 。

应用：串口转并口(内部时钟)

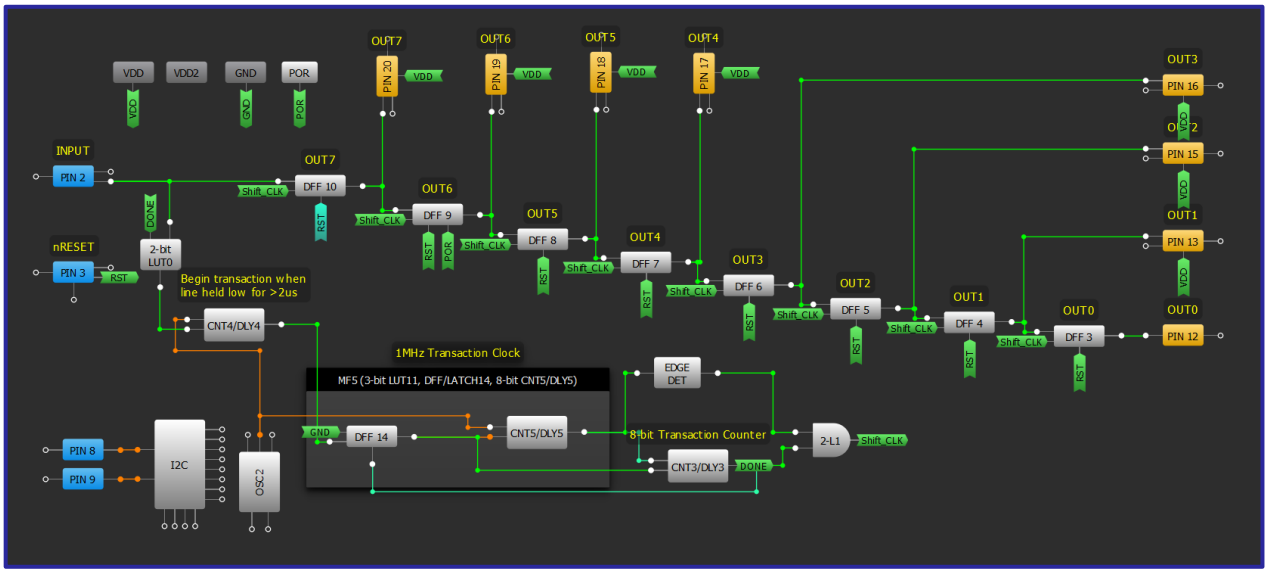
解串器[串口转并口]是将单线发送的串行多位数据由接收器解码为并行数据。当现实无法添加额外一条时钟线的时候，解串器可以作用为单线传输的功能。例如利用外部一个长达特定时间的低电平信号来触发 GreenPAK 内部的振荡器。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

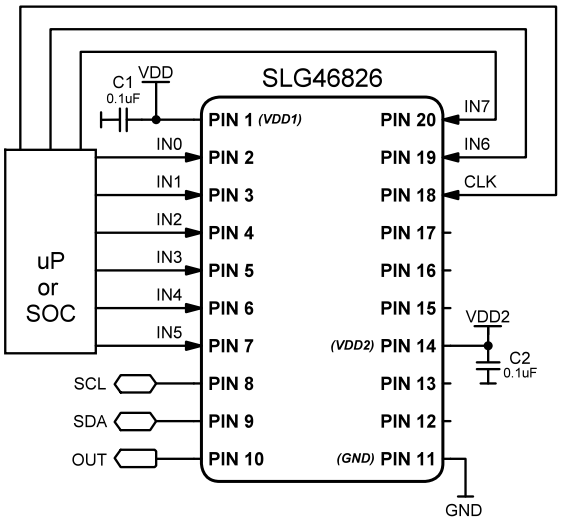


设计步骤

1. 使用 技巧：移位寄存器 配置好移位寄存器。
2. 为复位信号配置一个 **Digital input** 引脚，并将其连接到移位寄存器 **nRESET** [组成寄存器的 DFF **nRESET**]。
3. 使用 技巧：发送预设数量脉冲 配置好一个发送数量与移位寄存器位数相匹配的脉冲发生器。

应用：并口转串口

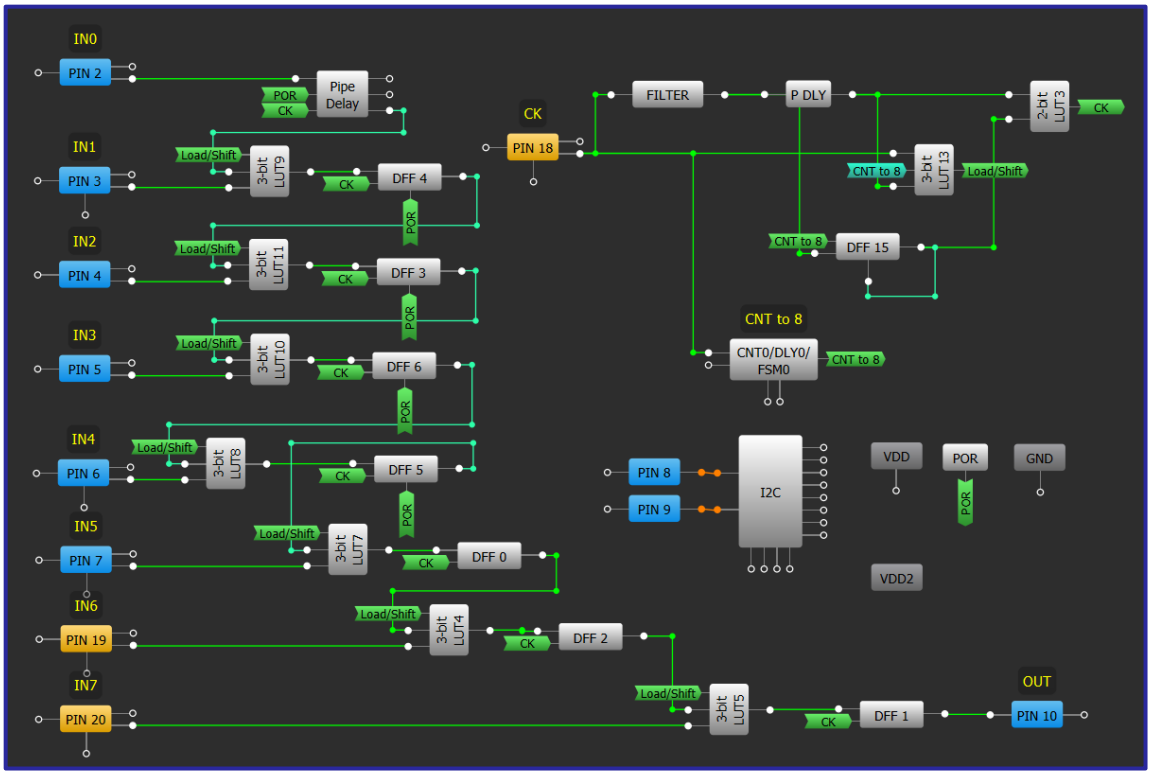
串行器[并口转串口]一般用于芯片间通过一根或者两根线[Data 和 Clock]进行数据传输，可以使用内部也可以使用外部时钟信号，并口的位数取决于GPAK 可用的I/O 和相关的内部资源。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

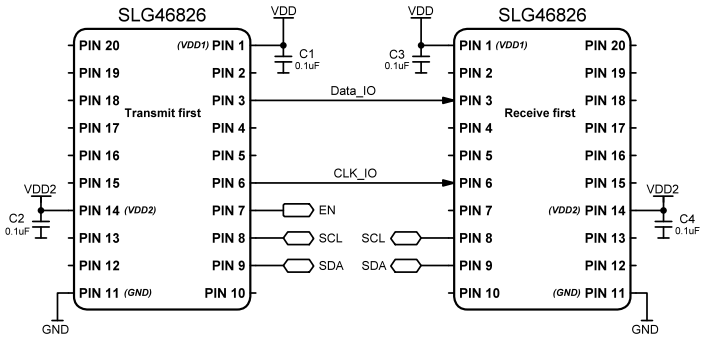


设计步骤

1. 使用 技巧：移位寄存器 配置好移位寄存器。
2. 使用 技巧：配置标准逻辑 为每个移位寄存器的 DFF 添加一个 3-bit LUT 并配置为 MUX。
3. 配置装载/移位切换功能，并将并行输入引脚和串行输出引脚连接到内部模块。

应用：双向通信 (发送优先)

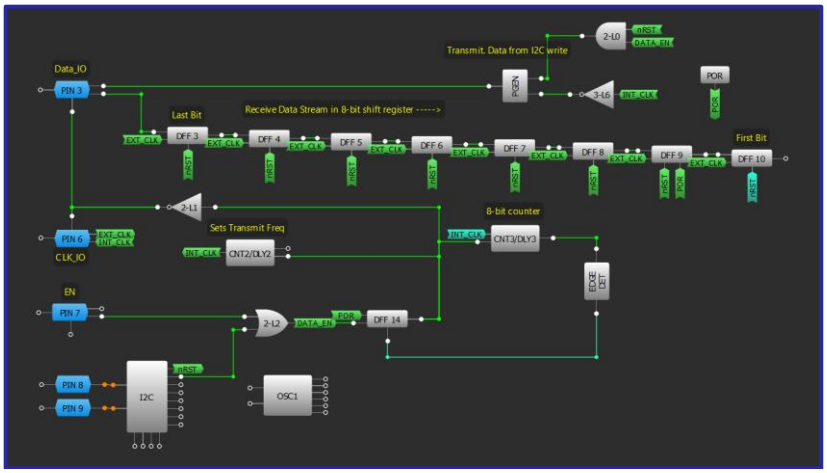
在主板空间不足或设备间（如可穿戴设备或充电器）相连的端口数目有限的设计中，双向通信系统显得尤为重要。发送优先设计中发送设备将始终在通讯事件中按第一优先级启用。



所需元器件

- 任意包含 OE 脚的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

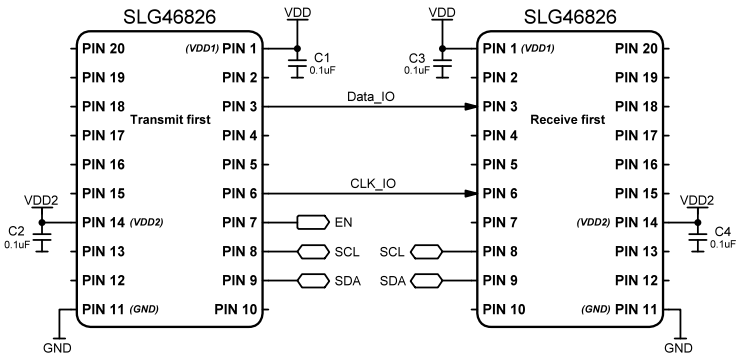


设计步骤

1. 将 2 个 GPIO 配置为 Digital Input/Output, 1 个为数据信号，1 个为时钟信号。
2. 将一个 CNT/DLY 模块配置为 **Reset counter** 以建立一个内部时钟信号。
3. 使用 技巧：移位寄存器 配置一个移位寄存器以存储传入数据。 I²C 可以读取移位寄存器的数据。
4. 添加 PGEN 以发送传出数据。I²C 可以改变 PGEN 的内容。
5. 配置 DFF 以使能数据传送。 将输出连接至传送时钟信号，并连接 输入/输出 的 OE 脚。
6. 添加 CNT/DLY 模块并配置为 **Reset counter** 以终止传输， 并选择内部时钟信号为 DFF 的输出。 其输出经过 Edge Det 模块后， 到达设定的时钟脉冲后将 DFF 复位。
7. 添加 CNT/DLY 模块并配置为 **Reset counter**, 并选择外部时钟信号作为输入，其输出用于使能传输 DFF。
8. 将外部时钟信号连接至移位寄存器，并将内部时钟信号连接至 **PGEN** 和 输入/输出脚。

应用：双向通信 (接收优先)

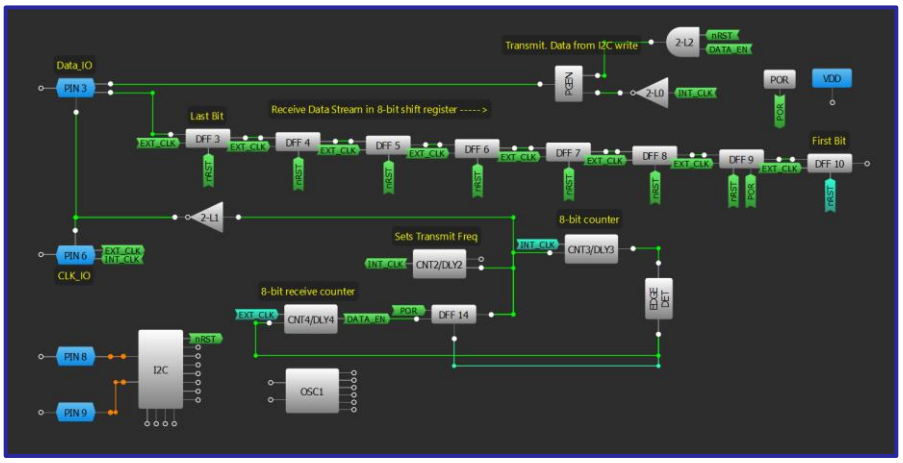
在主板空间不足或设备间（如可穿戴设备或充电器）相连的端口数目有限的设计中，双向通信系统显得尤为重要。接收优先设计中接收设备将始终在通讯事件中按第二优先级启用。



所需元器件

- 任意包含 OE 脚的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图

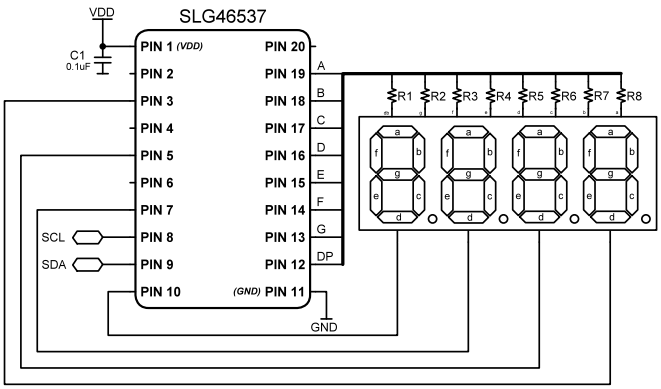


设计步骤

1. 将 2 个 GPIO 配置为 Digital Input/Output, 1 个为数据信号，1 个为时钟信号。
2. 将一个 CNT/DLY 模块配置为 **Reset counter** 以建立一个内部时钟信号。
3. 使用 技巧：移位寄存器 配置一个移位寄存器以存储传入数据。 I²C 可以读取移位寄存器的数据。
4. 添加 PGEN 以发送传出数据。I²C 可以改变 PGEN 的内容。
5. 配置 DFF 以使能数据传送。 将输出连接至传送时钟信号，并连接 输入/输出 的 OE 脚。
6. 添加 CNT/DLY 模块并配置为 **Reset counter** 以终止传输， 并选择内部时钟信号为 DFF 的输出，其输出经过 Edge Det 模块后，到达设定的时钟脉冲后将 DFF 复位。
7. 添加 CNT/DLY 模块并配置为 **Reset counter**, 并选择外部时钟信号作为输入，其输出用于使能传输 DFF。
8. 将外部时钟信号连接至移位寄存器，并将内部时钟信号连接至 **PGEN** 和 输入/输出脚。

应用：采用 ASM 和 I²C 的 7 段数码显示

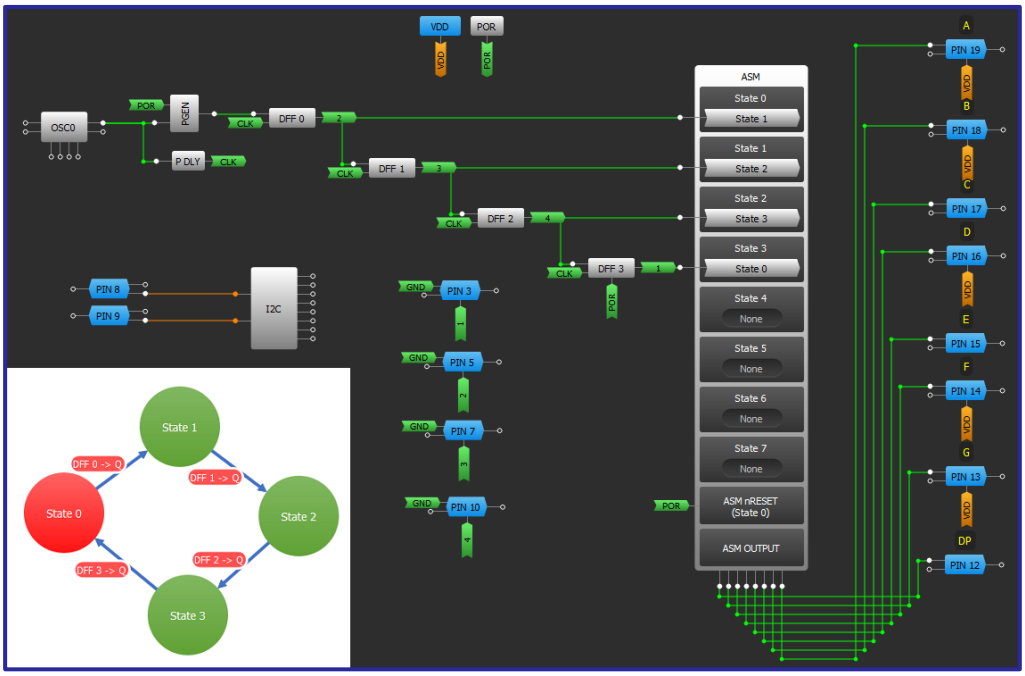
七段数码指示器是一种常见的数值显示装置。 GreenPAK 的异步状态机和I²C 可用于控制数码管进行数值显示。本实例展示了4 位包含小数位的十进制数字显示。



所需元器件

- 任意包含 I²C 和 ASM 的 GreenPAK IC(CMIC)
- 7 段数码显示管
- 8 个电阻

GreenPAK 设计图

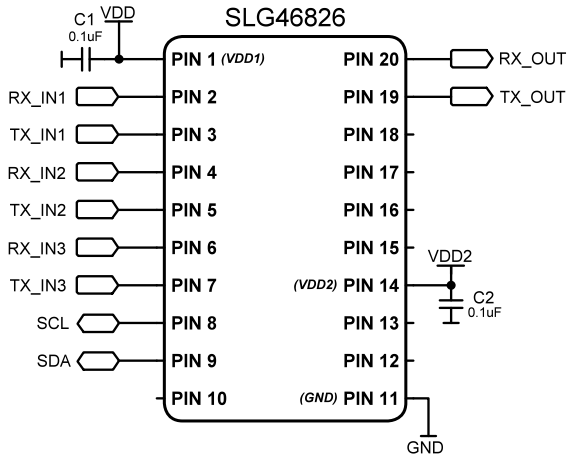


设计步骤

1. 将 GPIO 配置为 Digital Output 并连接到 ASM 的输出。
2. 使用 技巧：移位寄存器 添加一个移位寄存器。
3. 使用 PGEN 模块创建一个逻辑生成器，对应所需的位数。
4. 添加并配置异步状态机 (ASM) 以与初始数字序列相匹配。
5. 通过 I²C 将一个或多个数字更新到 ASM 状态中。

应用：采用 I²C 的通讯多路复用器

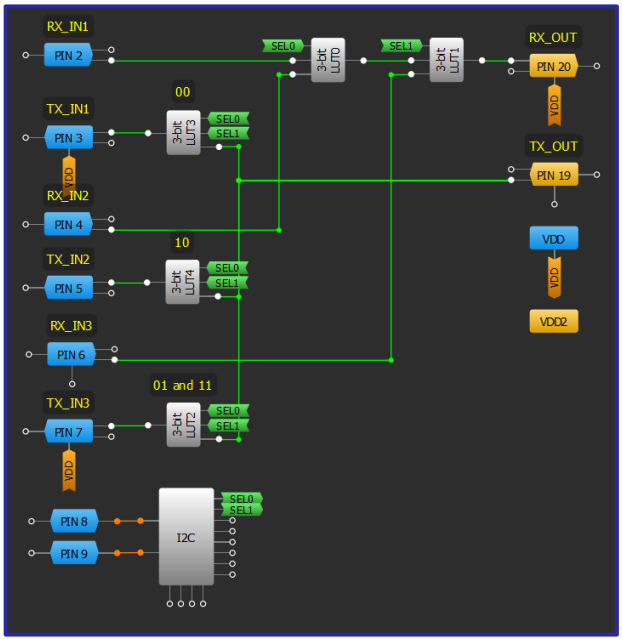
当设计人员需要整合几个不同设备的输入信号并将它们转发到一条输出线时，或者将单路输入信号通过指定的单个或者多个输出时，需要用到I²C 控制的通信多路复用器。



所需元器件

- 任意包含 I²C 的 GreenPAK IC(CMIC)
- 2 个电阻
- 2 个电容

GreenPAK 设计图

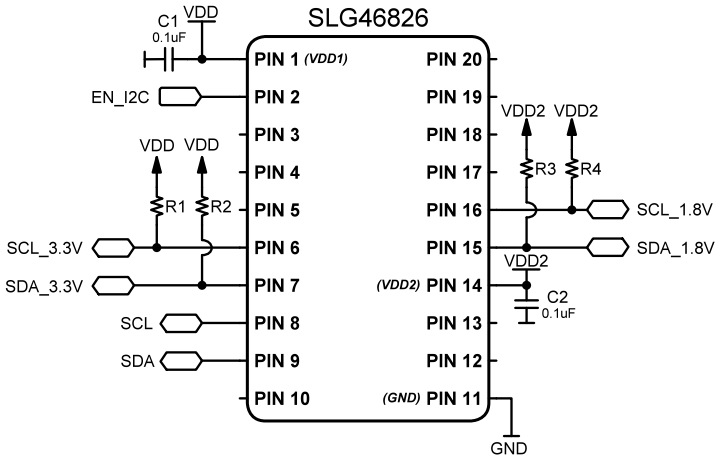


设计步骤

1. 用 LUT 配置 1：3 多路解复用器以共享来自 MCU 的 TX 信号，并发送至外部 UART 端口。
2. 用 LUT 配置 3：1 多路复用器以接收从外部 UART 端口到 MCU 的 RX 信号。
3. 使用 I²C 进行输入端口的切换。

应用：I²C 电平转换器

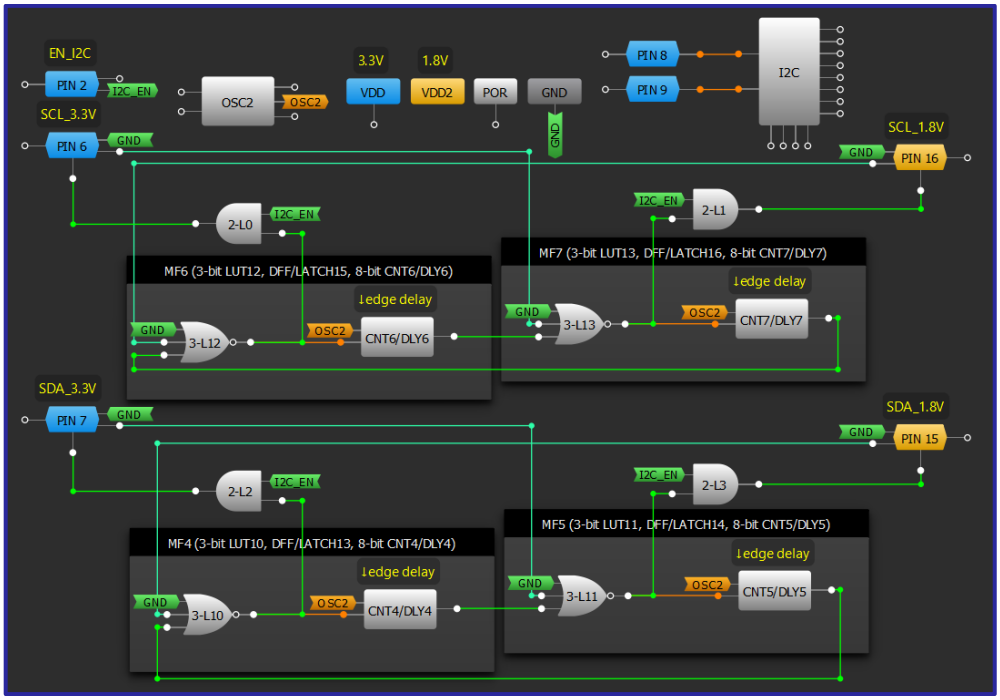
I²C 电平转换器允许两个含有I²C 的设备在两个不同电压下进行通讯。本实例展示了电压由 3.3V 转换为1.8V。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 4 个电阻

GreenPAK 设计图

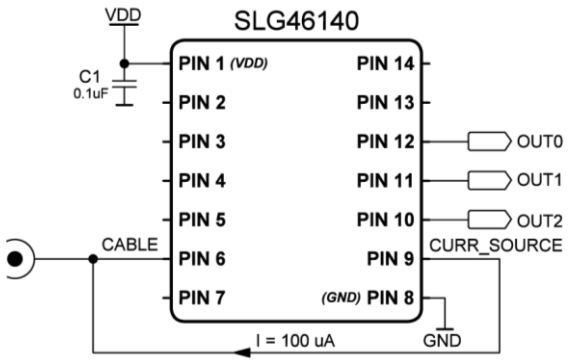


设计步骤

1. 将 GPIO 配置为 Digital Input/Output 并将 Output mode 设置为 open drain NMOS。
2. 将一个引脚配置为 Digital Input 作为使能信号。
3. 为每个输入/输出添加一个 AND gate (逻辑与门) 。
4. 将 4 个 Multi-function 组件(当缺少 Multi-function 组件时，可用 4 个 LUT 和 4 个 CNT/DLY 组件）配置为 Nor gate ， 接上一个下降沿延时。
5. 选择 **OSC2** 作为 DLY 组件的时钟源并设置为 **“Force Power On”**。

应用：连接检测

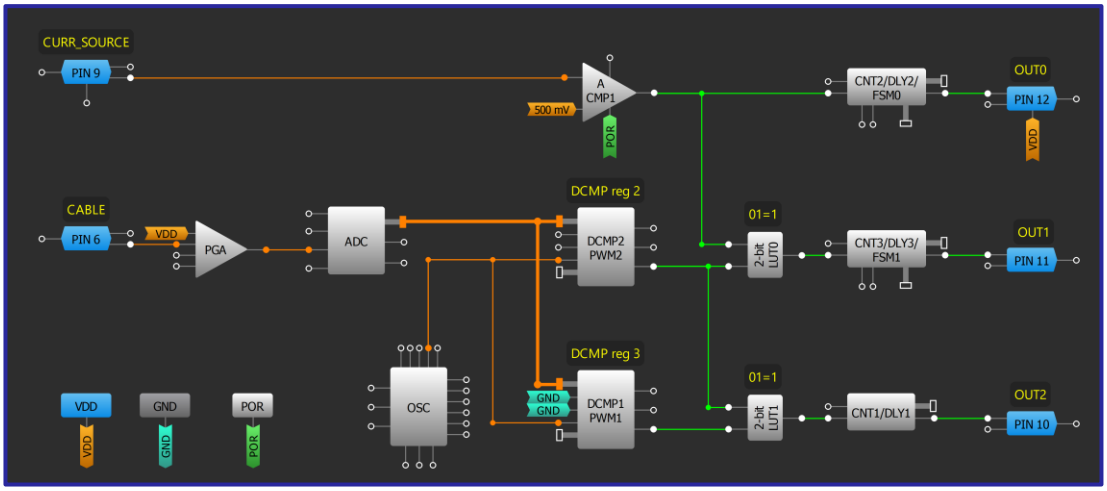
该应用程序通过测量与电阻成比例的电压来检测电缆的存在。ACMP 的 100uA 电流源用于在电缆上产生电压降。这种配置还能够根据不同的电线长度或连接负载来确定哪种类型的连接。



所需元器件

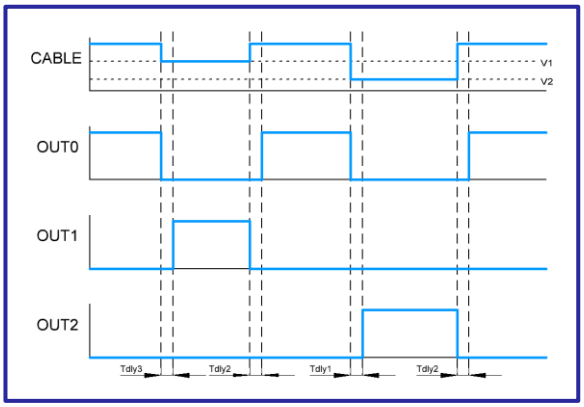
- 带 ADC 功能 GreenPAK IC(CMIC)

GreenPAK 设计图



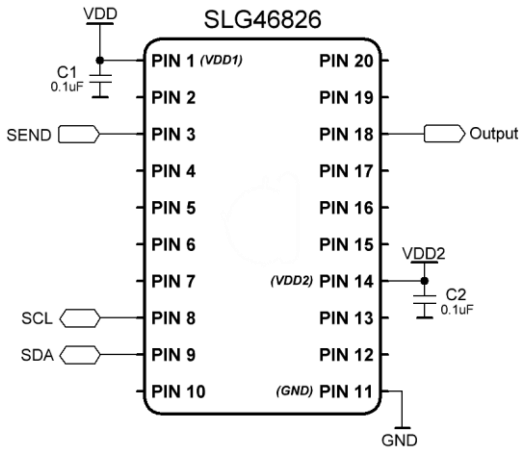
设计步骤

1. 启用 **ACMP1** 中的 **input 100uA current source**，并配置 **IN- source**。
2. 配置 ADC 和 PGA 块。
3. 使 DCMP/PWM 模块(从 **SHARED PD** 输入端子上移除 VDD)。将 DCMP/PWM Power 寄存器设置为 Power on。确保 **IN+ selector** 连接到 ADC, **IN- selector** 器连接到内部寄存器。
4. 在 DCMP/PWM 模块中设置寄存器所需的值，并配置 **MTRX SEL** 输入。
5. 配置 **2-bit LUT0** 和 **2-bit LUT1**。
6. 设置 **CNT1-CNT3** 为上升沿延时模式。
7. 配置 **PIN10-PIN12** 为数字输出 1x push pull。



应用：自定义模式发生器

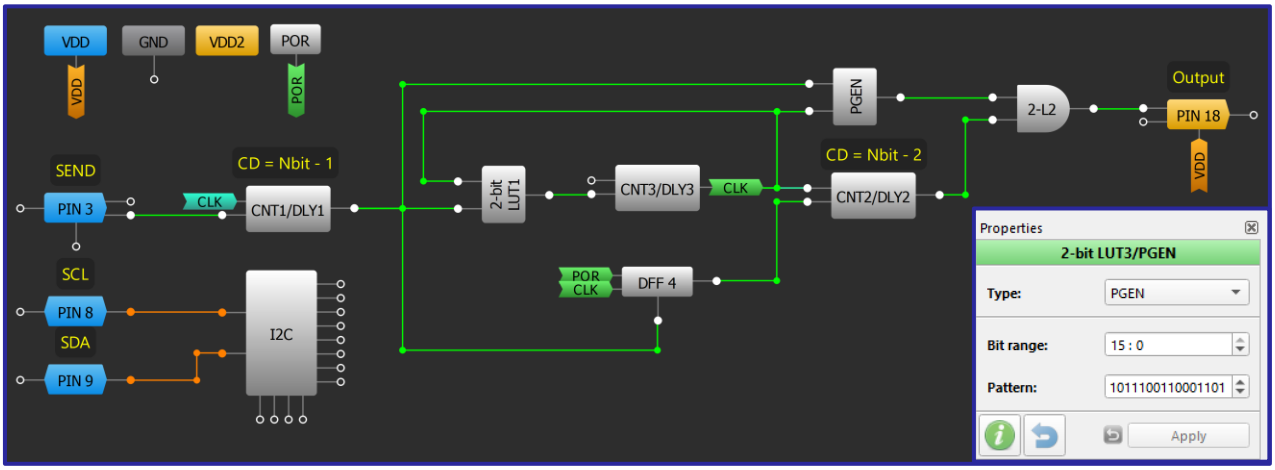
GreePAK 中的模式发生器(PGEN)存储一组逻辑 1 和 0(最多 16 位)的数据组，该数据组可以串行地发送到内部矩阵。这个设计在输入上的上升沿之后输出一个 16 位的代码。它可以用于顺序逻辑。



所需元器件

- 任意带 PGEN 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



设计步骤

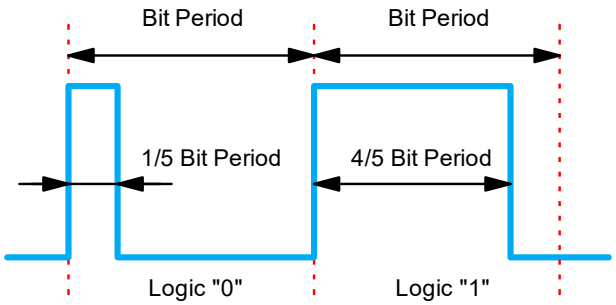
1. 在 **PGEN** 中配置自定义 N_bits 模式。
2. 将 **CNT1/DLY1** 模式配置为 One Shot 模式。设置 Counter Data 为“$Counter\ Data = N_{bits} - 1$.
3. **“CNT2/DLY2”**模式配置为“One Shot”模式，“Counter Data” 设置为“$Counter\ Data = N_{bits} - 2$.
4. 将 **CNT3/DLY3** 配置为上升沿延迟，使用 **2 位 LUT1** 创建发生器。
5. 通过使用带有 **I²C** 的 GreenPAK 来动态修改 PGEN 数据、时钟频率和计数器数据，这样可以改进设计。

技巧：使用占空比检测发送串行协议

此技巧适用于任何带有模式发生器、计数器/延时器和I²C 的GreenPAK 芯片。请阅读 技巧：使用移位寄存器读取串行协议 和 技巧：使用管道延时(Pipe Delay)读取串行协议 以了解此处所讨论的如何利用GreenPAK 芯片读取协议。

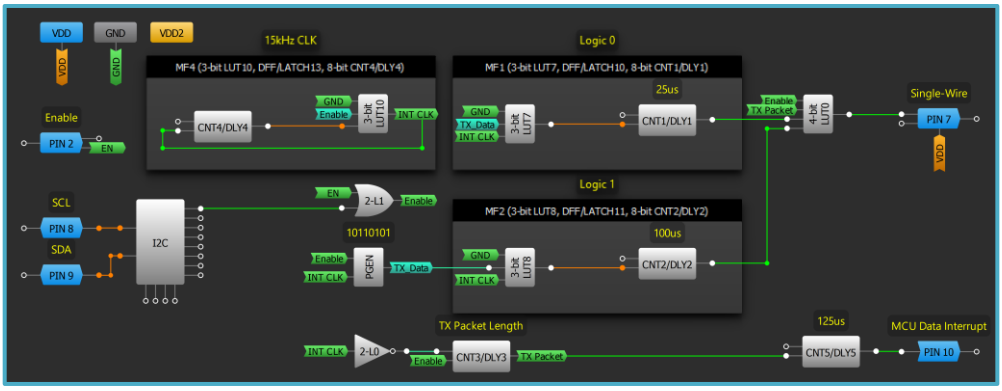
串行数据传输具有非常广泛的通讯应用包括电力线、无线系统和单片机系统。单线数据传输可以使用不同的占空比来传输数据，以供另一个 GreenPAK 或 MCU 读取。

占空比检测技术如下图所示。每一个数据位由一个特定占空比的周期脉冲来表示。在此设计中，当脉冲持续时间≤整个周期的 1/5 时，传输位被设置为“0”，当脉冲持续时间≥整个周期的 4/5 时，被设置为“1”。可以使用其他占空比范围，前提是逻辑“0”和逻辑“1”的占空比范围之间有足够的区分。



占空比检测技术 -- 逻辑 0 和逻辑 1

在如下所示的设计中，数据传输由 **Enable** 脚发起，这个操作可以通过一个外部的 GPIO 或者一个 I²C 虚拟输入实现。**CNT4/DLY4** 设置数据传输的位周期。**MF1** 和 **MF2** 模块定义传输的占空比。传输的数据通过 I²C 或非易失性存储器写入 PGEN。PGEN 最多可传输 16 位数据，而该设计展示了单线输出上的 8 位数据传输。有关更多如何利用 I²C 通过 PGEN 生成不同的模式，请参阅应用：自定义模式发生器。



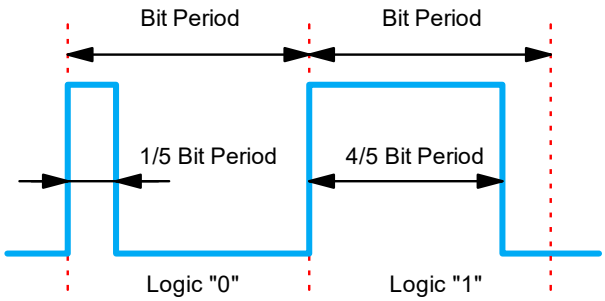
数据包长度由 **CNT3/DLY3** 确定，在本实例中它被设置为传输 8 位数据。数据嵌入到传输信号的占空比后，**4 位 LUT0** 通过 **Single-Wire** 输出脚传输数据。当数据传输完成后，可以通过 I²C 对 PGEN 重写。

如果用户希望中断传输，CNT5/DLY5 会产生一个中断信号，向外部 MCU 表明传输包已经完成。

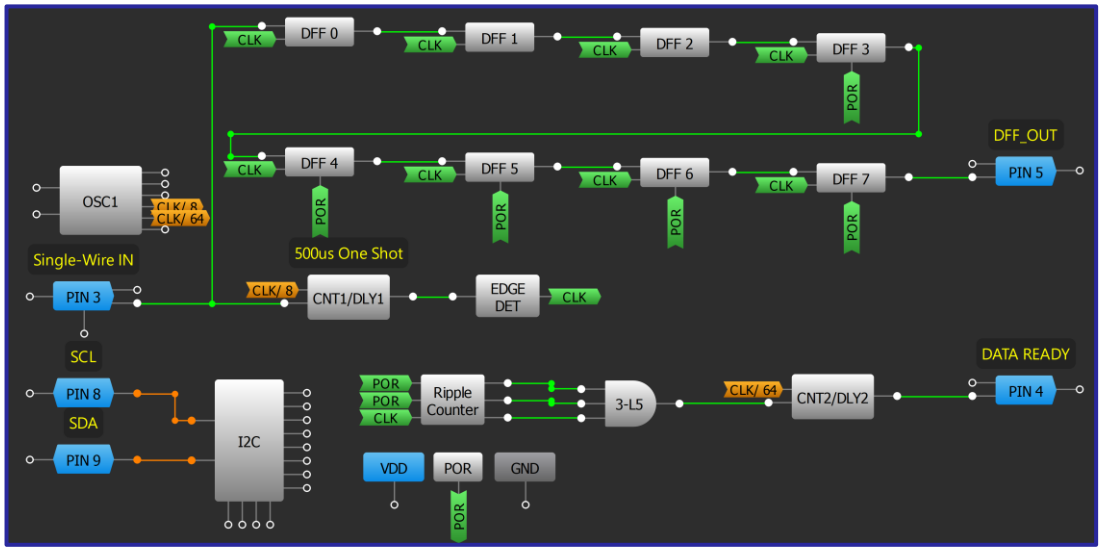
技巧：使用移位寄存器读取串行协议

此技巧适用于所有带有足够的D 类触发器、1 个Ripple Counter 和3 个计数器/延时器的GreenPAK 芯片。请阅读技巧：使用占空比检测发送串行协议 以了解一种读取单线协议的方法。

串行数据传输具有非常广泛的通讯应用包括电力线、无线系统和单片机系统。移位寄存器可以用作单线传输中高度通用的串行接收器。



上图显示了在单线拓扑中所讨论的技巧:使用占空比检测发送串行协议。上升沿上的延迟边沿检测用于在信号周期（位长度）的一半进行采样。对这种特殊设计的移位寄存器记录数据时，当脉冲宽度≤1/5 位周期时信号为低电平，当脉冲宽度≥4/5 位周期时，信号为高电平。



上图显示了一种以 8 位增量获取单线数据的方法。每次对移位寄存器输入移位时钟，**Ripple Counter** 也会递增。当 **Ripple Counter** 收到 8 个脉冲，**CNT2/DLY2** 在 **DATA READY** 脚输出一个高脉冲信号以通知 MCU 已读取完整的单线传输数据并准备好由 I²C 读取。本设计中，由于 CNT1/DLY1 将上升沿延迟 500μs，因此该设计以 1ms 周期读取传输数据，但这可以通过调整其计数器值轻松改变。

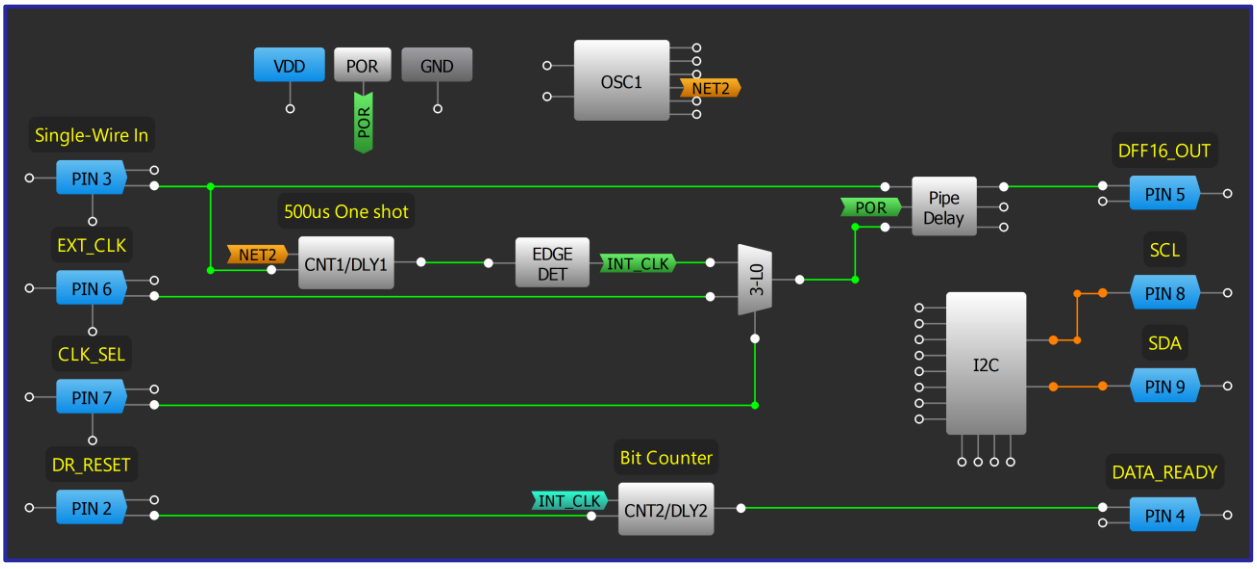
技巧：利用管道延时(Pipe Delay)读取串行协议

此技巧适用于所有GreenPAK 芯片。在技巧：使用占空比检测发送串行协议中介绍了一种读取单线协议的方法。

串行数据传输具有非常广泛的通讯应用，包括电力系统、无线系统和单片机系统。可以使用单个 Pipe Delay 模块代替资源繁重的移位寄存器，作为单线传输的串行接收器。但需要注意，与移位寄存器方式不同的是，这种 Pipe Delay 方式只能串行输出其数据，而不具备并行输出的能力。

可以把 Pipe Delay 想象成一个有 3 个可用输出的 16 位移位寄存器。其中 2 个输出可配置为 Pipe Delay 里 16 个内部 DEF 输出中的任何一个。对于这个 16 位的输出，如下图所示，Pipe Delay 的 **OUT0** 设置为代表第 16 个 DFF 输出。 MCU 用于为 Pipe Delay 提供外部时钟的输入与 **Single-Wire In** 输入复用，以允许它在不增加位计数器的情况下卸载 Pipe Delay 的数据。

本设计采用了与技巧：使用移位寄存器读取串行协议 相同的延时数据采样机制，但它使用计数器模块而不是异位计数器来跟踪其数据位。每检测到一个数据位时，该计数器都会递增。当达到 16 位时，它将 **DATA_READY** 引脚设置为高电平，直到发送下一个数据位。很重要的一点是，在发送任何数据之前，必须重置这个计数器模块，以确保正确的操作，这可以由 POR 在内部完成，也可以由一个输入在外部完成(如本设计中所示)。



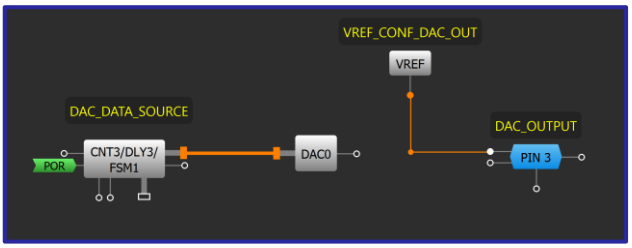
技巧：数模转换器

此技巧适用于 SLG46140,SLG46620 和 SLG46621 GreenPAK 芯片。

有些 GreenPAK 芯片包含**数模转换器(DAC)**。它们是 8 位、最大采样速度为 100ksps 的数模转换器。其微分非线性小于 1LSB，积分非线性小于 1LSB。DAC 输出至 pin 的电阻是 1 kΩ。建议负载电阻不小于 10kΩ，负载电容不大于 100pf。通常，DAC 输出范围为 0V ~ 1V，但在 SLG46620/1 中，DAC1 输出范围为 50mV ~ 1.05V。

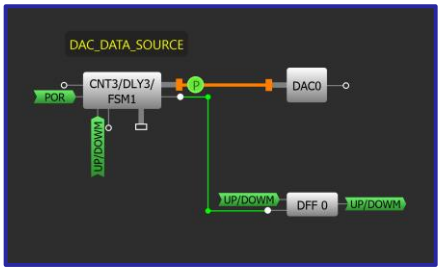
寄存器、SPI 或 FSM 均可配置为 DAC 的输入。 DAC 的输出可以配置为 VREF 的输出引脚、PGA 或 ACMP。

在某些 IC 中，**DAC0** 用作 PGA 宏单元伪差分模式的一部分。因此，当 PGA 处于伪差分模式时，**DAC0** 不可用。此外，**DAC1** 与 ADC 宏单元复用。因此，使用 ADC 时不能使用 **DAC1**。要将 DAC 宏单元输出连接到 VREF 宏单元，必须将此引脚配置为 analog input/output，并将 VREF 的 **Source selector** 配置为 DAC。



锯齿波发生器

DAC 可用于创建如上所示的简单锯齿波发生器。在这种情况下，**DAC0** 使用 **FSM1** 作为其 data source。DAC output 连接到 VREF，VREF 连接到 **PIN3 (DAC_OUTPUT)**。输出信号周期和分辨率由 **FSM1** 的 counter data 和时钟频率设置。此外，您可以添加一个 toggling DFF 并将其连接到 FSM 的 **UP** 输入以产生如下所示的三角波发生器。counter data 值随时间上下变化，取决于 **DFF0** 的输出。



三角波发生器

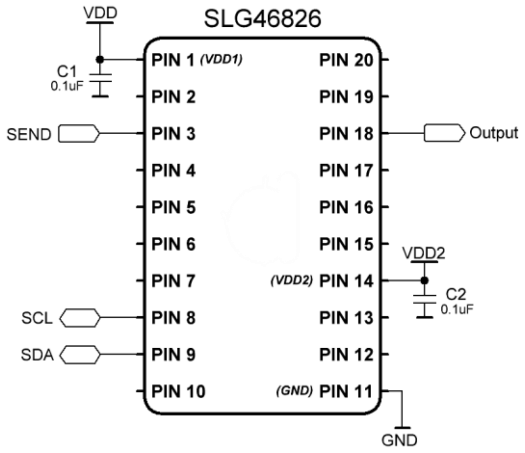
DAC 可用作“差分”和“伪差分”模式下 ACMP 或 PGA 负输入端的基准。如果 ACMP 的基准使用 DAC，则用户可以将模拟信号与离散数据进行比较，而无需使用 ADC 模块。

GreenPAK DAC：

- 可用于创建波形发生器；
- 创建通过 SPI 控制的基准电压源；
- 可用于不同的转换器，因为它将温度、湿度和其他数字化值转换为模拟电压。

应用：自定义模式发生器

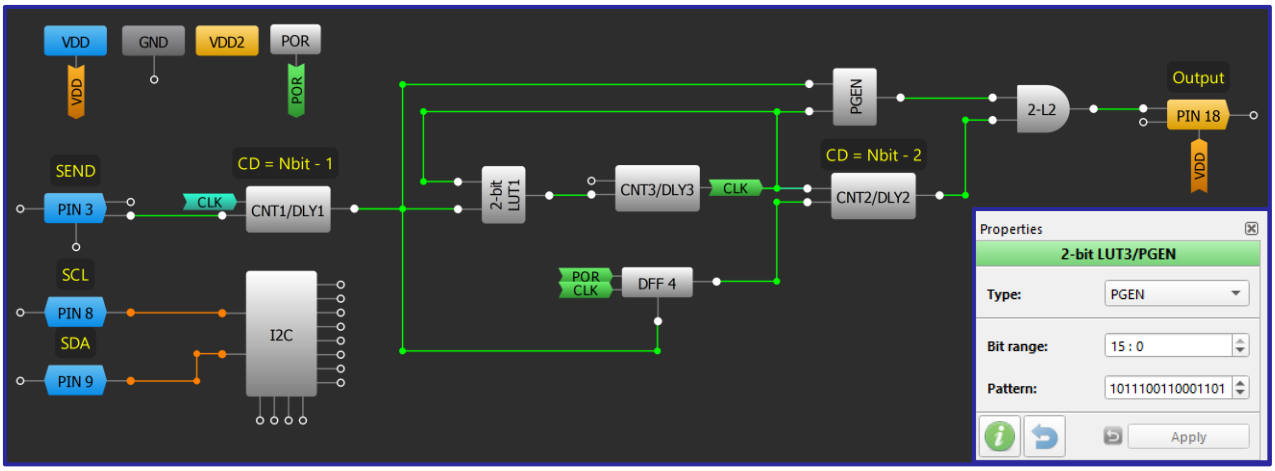
GreePAK 中的模式发生器(PGEN)存储一组逻辑 1 和 0(最多 16 位)的数据组，该数据组可以串行地发送到内部矩阵。这个设计在输入上的上升沿之后输出一个 16 位的代码。它可以用于顺序逻辑。



所需元器件

- 任意带 PGEN 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



设计步骤

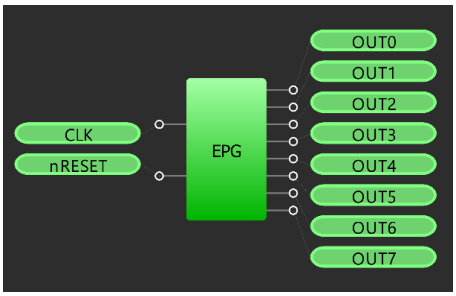
1. 在 **PGEN** 中配置自定义 N_bits 模式。
2. 将 **CNT1/DLY1** 模式配置为 One Shot 模式。设置 Counter Data 为“$Counter\ Data = N_{bits} - 1$.
3. **“CNT2/DLY2”**模式配置为“One Shot”模式，“Counter Data” 设置为“$Counter\ Data = N_{bits} - 2$.
4. 将 **CNT3/DLY3** 配置为上升沿延迟，使用 **2 位 LUT1** 创建发生器。
5. 通过使用带有 **I²C** 的 GreenPAK 来动态修改 PGEN 数据、时钟频率和计数器数据，这样可以改进设计。

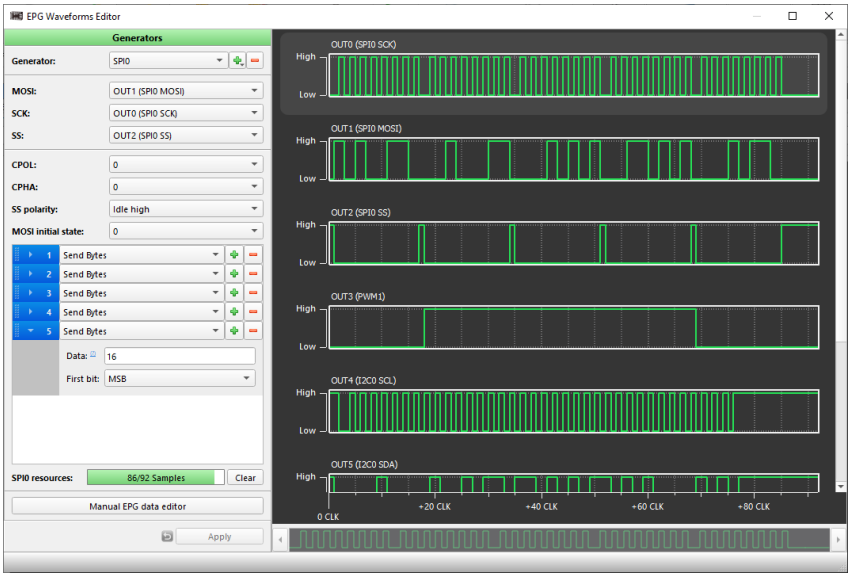
技巧： EPG

扩展模式/序列发生器 (EPG) 能够产生一系列 92 字节(92 Bytes)长度的输出。 它从非易失性存储器 (NVM) 检索数据，并在每个时钟输入信号的上升沿输出一个字节。此外，EPG 与 I²C Virtual Inputs 共享其输出端子(OUT0~7)。最大时钟频率可达 1 MHz。

系统上电后，EPG 根据 nReset 引脚收到的信号，呈现不同的行为。当 nReset 输入为有效低电平时，EPG 将输出初始值。相反地，当 nReset 输入为有效高电平时，EPG 将输出用户自定义的模式。此功能使用户能够根据自己的特定需求定制 EPG 的输出。

当 CLK(CNT overflow/keep) 设置为 Overflow 模式时，EPG 可以连续工作，或者在 Stop at boundary 模式下保持最后一个字节。
EPG 波形编辑器允许用户从各种预定义的发生器(Generator 下拉项目 SPI、I²C、PWM 或手动配置）中进行选择，并相应地配置输出。每个发生器的预定义配置都提供一系列可调设置，使用户更容易构建他们期望的模式。

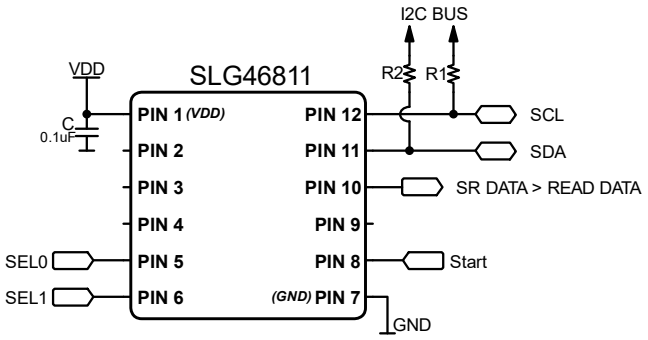




资源栏呈现可视化展示，指示模式/序列中所使用的位数。它使用户可以清楚地了解已分配给当前模式/序列的内存和资源量。

应用：带 ACK 检查和数据比较的 I²C 主设备读指令

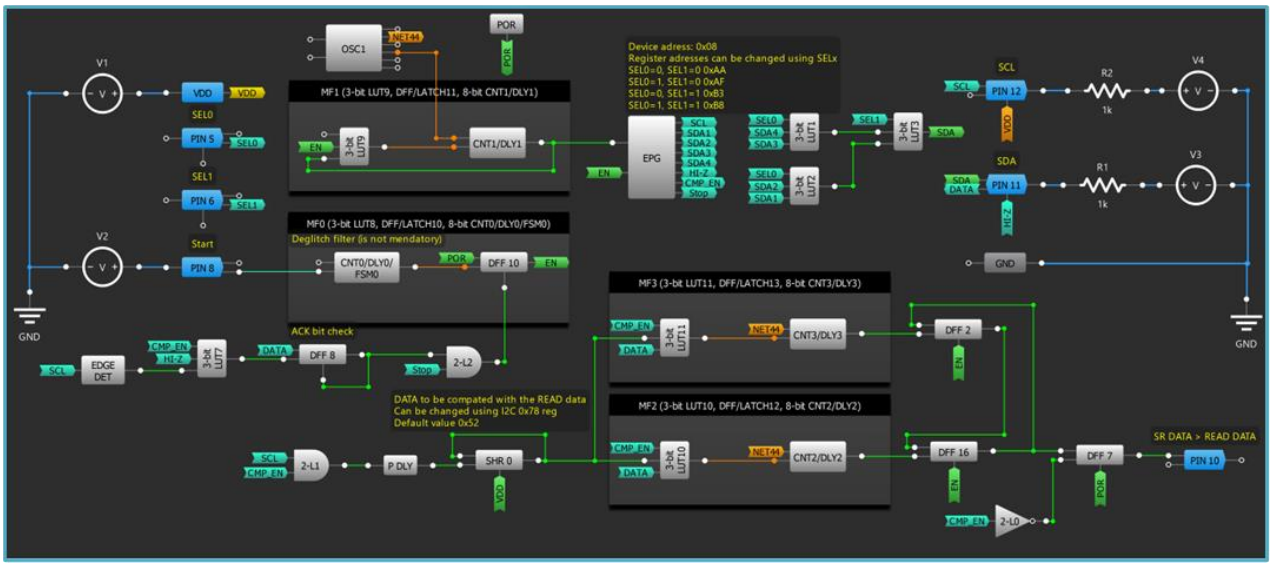
本实例展示了如何利用 SLG46811 构建一个简单的 I²C 主设备，该主设备可以读取从设备数据并将其与预设好的参考数据进行比较。



所需元器件

- SLG46811V
- 2 个电阻器

GreenPAK 设计图

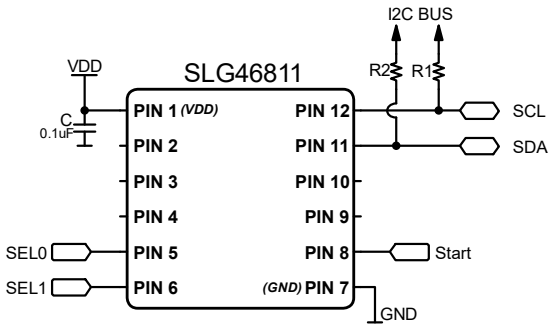


设计步骤

1. 配置 EPG 发生器（请参阅 技巧：EPG）以创建 I²C 读命令并设置多种 SDA 和 SCL 模式。
2. 在 EPG 中添加一个通道，用于每 9 个 SCL 时钟周期进行一次的 ACK 位校验。
3. 创建一个 4 位多路复用器，参考应用：8 位多路复用器。SEL0 和 SEL1 用于选择要输出的数据(SDA)。
4. 基于 CNT1/DLY1 和 LUT9 创建一个由 EN 信号控制的时钟源，为 EPG 提供时钟。
5. 通过 CNT0/DLY0 去抖/滤波后的信号触发 EN 进入 I²C 模式。
6. DFF8 用于监控 I²C 指令传输期间是否存在 ACK 位，如果 I²C 从设备没有响应，则传输停止。
7. 当 I²C 从设备开始发回数据时，它们将与存储在 SHR 0 中的参考数据进行比较。在比较过程中，SHR0 中的数据会随着 SCL 移位并与接收到的数据逐位进行比较。 DFF7 存储比较结果直到接收到读命令为止。

应用：带 ACK 检查的 I²C 主设备写指令

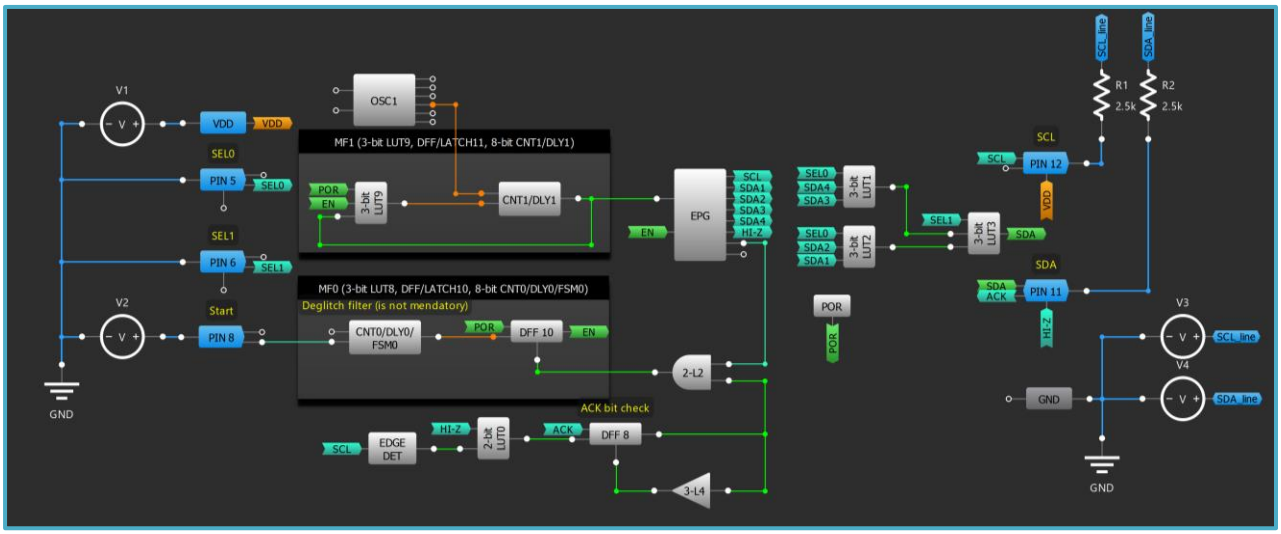
本案例展示了通过 SLG46811，构建一个简单的 I²C 主设备。利用该芯片创建的 I²C 主设备能够重复写入一个或多个字节，这使其成为无需复杂设计的系统的绝佳解决方案。



所需元器件

- SLG46811V
- 2 个电阻器

GreenPAK 设计图

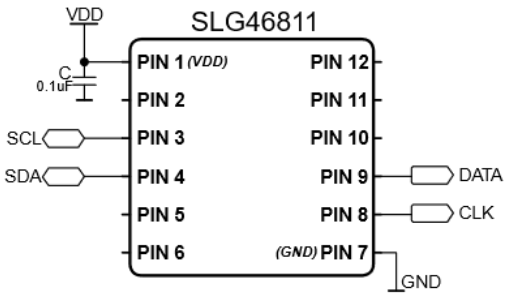


设计步骤

1. 配置 EPG 发生器（请参阅 技巧：EPG）以创建 I²C 写命令并设置多种 SDA 和 SCL 模式。
2. 在 EPG 中增加一个通道，用于 SCL 时钟的第 9 个时钟进行一次的 ACK 位校验。
3. 创建一个 4 位多路复用器，参考应用：8 位多路复用器。SEL0 和 SEL1 选择要输出的数据(SDA)。
4. 基于 CNT1/DLY1 和 LUT9 创建时钟源，为 EPG 提供时钟。
5. 通过 CNT0/DLY0 去抖/滤波的信号可触发使能 I²C 模式。
6. DFF8 用于监控 I²C 指令传输期间是否存在 ACK 位，如果 I²C 从设备没有响应，则传输停止。

应用：移位寄存器构建 I²C 可编程模式发生器

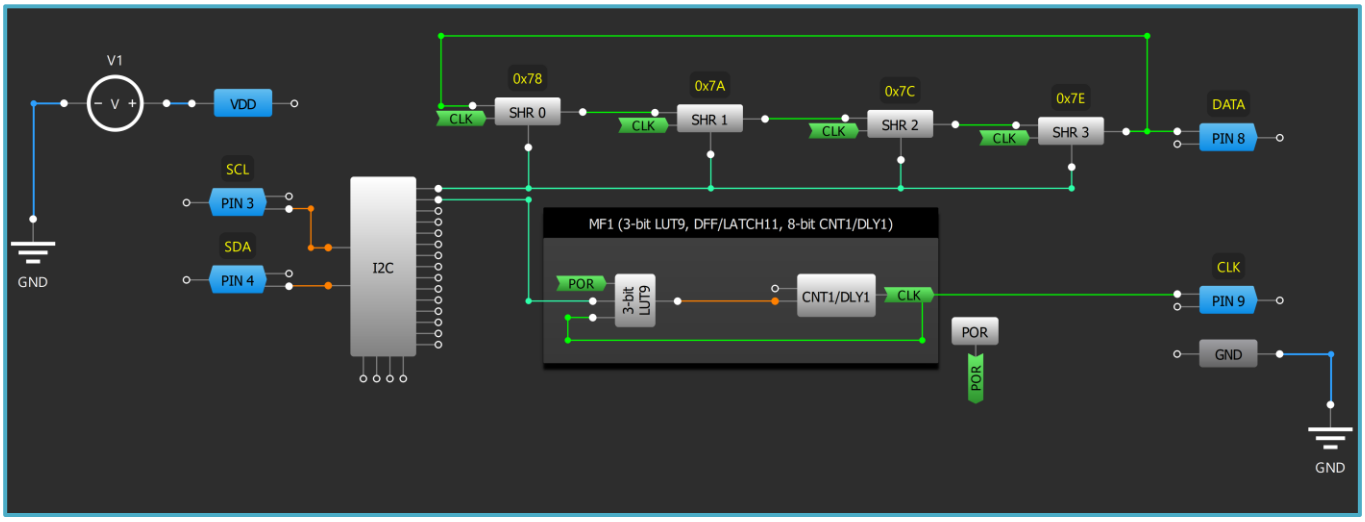
本实例展示了利用 SLG46811 构建一个简单的、容量高达 32 位的移位模式发生器。这是一种经济高效的应用。



所需元器件

- SLG46811V 或者任意带有移位寄存器的 GreenPAK 芯片

GreenPAK 设计图

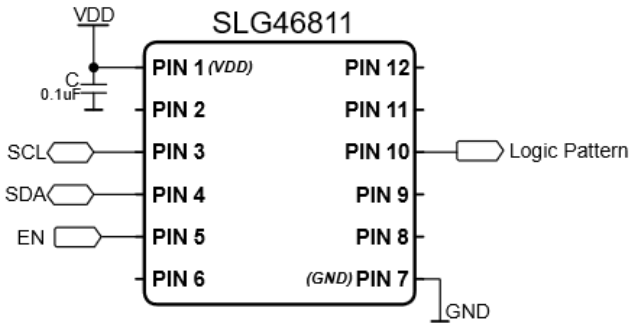


设计步骤

1. 配置移位寄存器模块并将它们串行连接。
2. 将 SHR3 的输出连接到 SHR0 的 D 输入脚。
3. 使用 MF1 中 CNT1/DLY1 和 LUT9 创建时钟源。
4. 使用 I²C virtual inputs 清零 SHR0-SHR3，之后再次写入 32bit 的移位数据并开始发送。
5. 增加 PIN9 并连接至 CLK 输出同步时钟用于识别数据。

应用：固定长码发生器

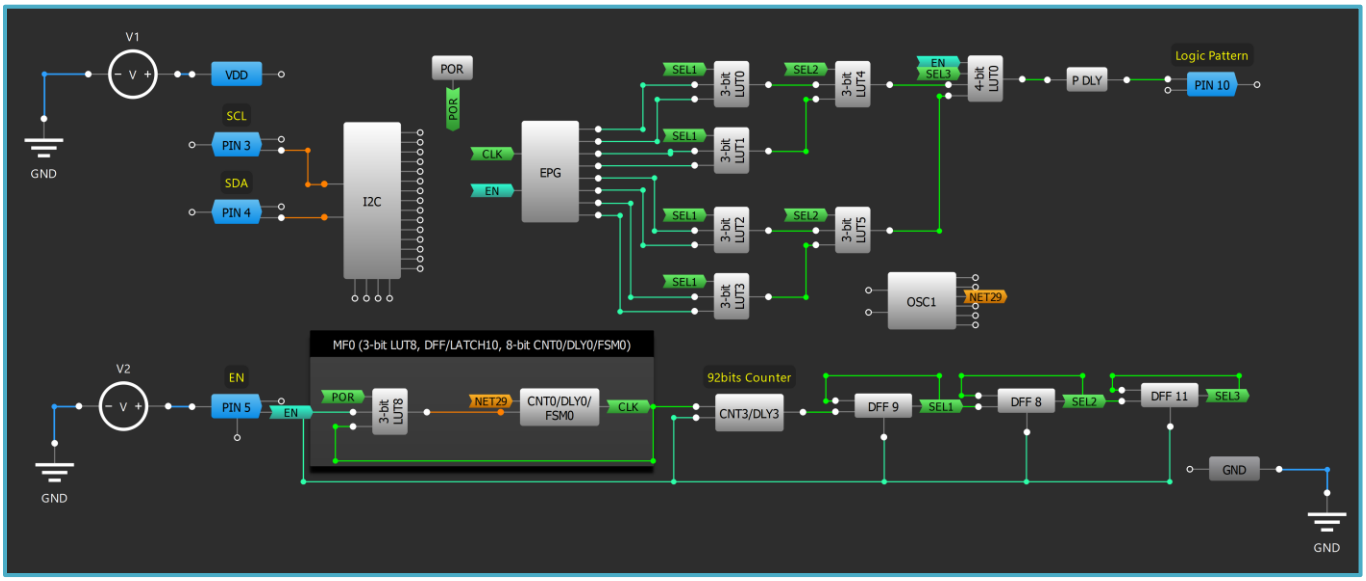
利用SLG46811 的EPG 宏单元构造一个容量高达736 位的模式发生器，低成本和低功耗的优点使其可以应用在许多实际场景下。



所需元器件

- SLG46811V

GreenPAK 设计图

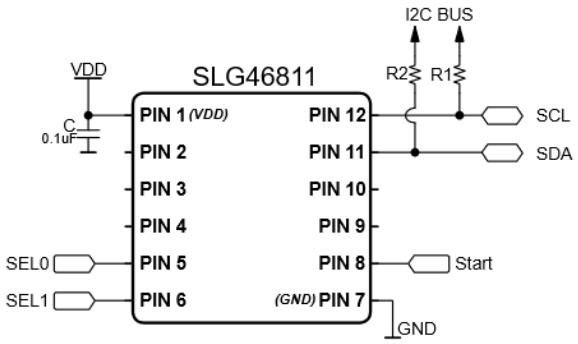


设计步骤

1. 配置 EPG 发生器(参见技巧：EPG)，设置固定模式输出的数据 。
2. 创建一个 8 位多路复用器，如参考应用：8 位多路复用器，并连接到适当的 EPG 输出 。
3. 创建一个基于 CNT0/DLY0 和 LUT0 的频率发生器，为 EPG 产生输入时钟。
4. 配置 CNT3/DLY3 用来计数频率发生器产生的 92 个时钟。
5. 基于 DFF8、DFF9 和 DFF11 的 3 位计数器用于选择从 EPG 输出的数据，参见技巧：多路复用比特流
6. 添加 EN 从 PIN5 输入，高/低电平分别使能/关断 EPG。

应用：简单 SPI 主机

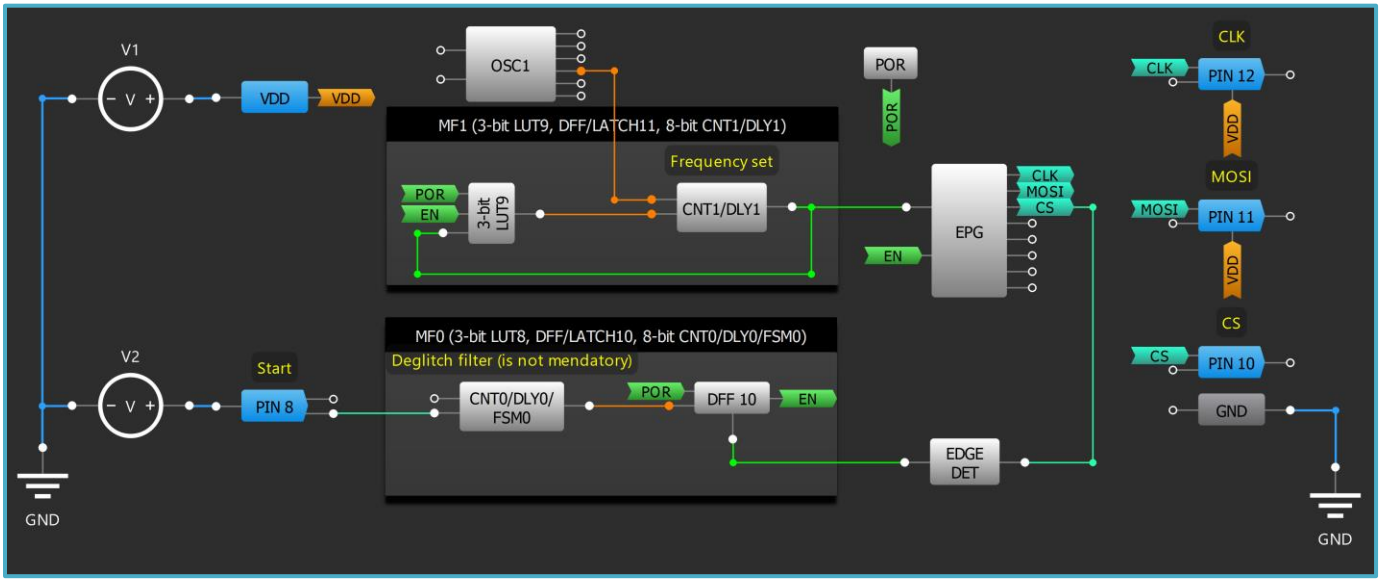
这个应用演示了如何使用SLG46811 设备构建一个简单的SPI 主控器。使用该设备的SPI 主控器能够重复输出一个或多个字节，对于一些不需要复杂功能的系统提供了有效的解决方案。



所需元器件

- SLG46811V

GreenPAK 设计图



设计步骤

1. 配置 EPG 发生器(参见 EPG technique)来创建 SPI 发生器(Generator 选择为 SPI)，并选择相应的 MOSI, SCL, CS 信号输出。
2. 创建基于 CNT1/DLY1 和 LUT-9 的频率发生器，为 EPG 提供输入时钟。
3. CNT0/DLY0 构成 Deglitch 滤波器，被滤波的信号会通过 DFF10 产生 EN 信号使能 SPI 主机模式。
4. 使用 EDGE DET 为 Rising edge detect 并配置成 nOUT 模式以复位 DFF10。当 CS 重新变高时，EDGE DET 产生短暂的低电平以复位 DFF10，EN 信号因此变低。

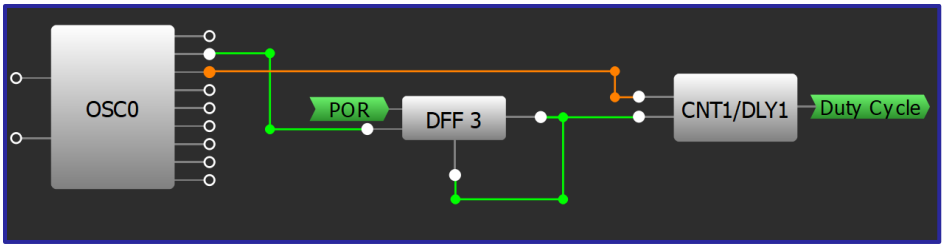
第 6 章: 基于脉冲的控制

本章介绍了控制信号脉冲宽度的应用。这些应用常常涉及 PWM 控制，通常用于 LED 控制器，电机控制器，声音控制器。

技巧：恒定占空比 PWM

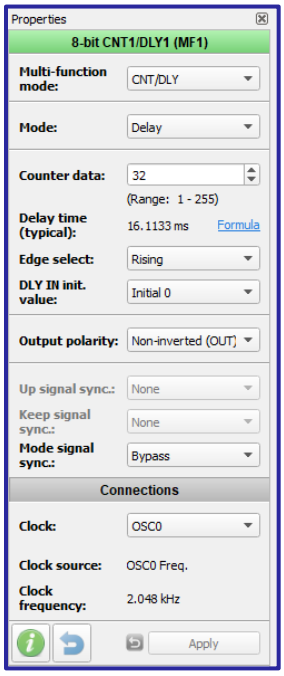
此技巧适用于所有 GreenPAK。

设定一个固定占空比的 PWM 信号需要一个 CNT/DLY，一个振荡器和一个 DFF，设计图如下图所示。



简单的占空比设计图

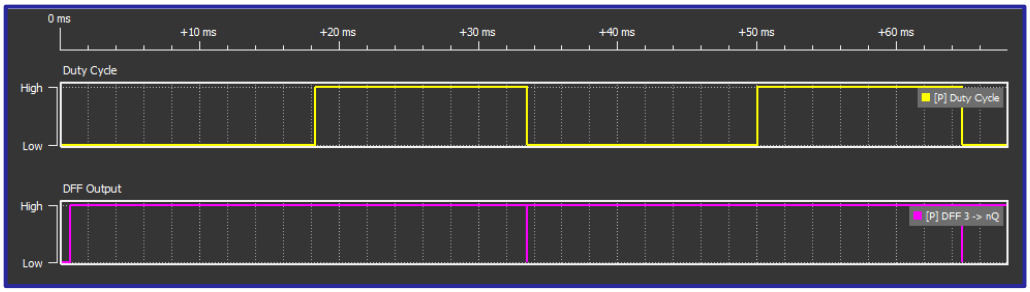
DFF 作为一个上升沿检测器，DFF 的 CLK 信号(或者图中振荡器 OSC 的输出时钟)确定 PWM 信号的周期， CNT/DLY 如 Figure 26 所示配置为上升沿 Delay，决定 PWM 信号的占空比，当 DFF 的 CLK 接收到上升沿，将输出一个低电平的脉冲到 CNT1/DLY1，由于 CNT1/DLY1 配置为上升沿 Delay，CNT1/DLY1 将输出低电平， 并在设定的延时时间后输出高。



CNT/DLY 配置

将 DFF 的 **Q output polarity** 配置为 **Inverted (nQ)** ，并且连接 DFF 的 **nQ** 连到 **nRESET**，这样 DFF 将工作为一个上升沿检测器，除了 CLK 检测到上升沿时输出一个短时低电平脉冲外，其余时间将维持在高电平， **FILTER/EDGE DET** 组件也可以配置为类似的功能。

可以用振荡器的 **OUT0** 或者 **OUT1** 作为 DFF 的 CLK 以确定输出 PWM 的周期，周期应当始终大于高电平时间，在这个例子当中， **'OUT1' second divider by** 设置为 **64**，因此周期为 Tosc*64 CNT/DLY 的 **Counter data** 设定 PWM 低电平时间，PWM 的占空比计算公式： $D = \frac{T_{period} - T_{delay}}{T_{period}}$

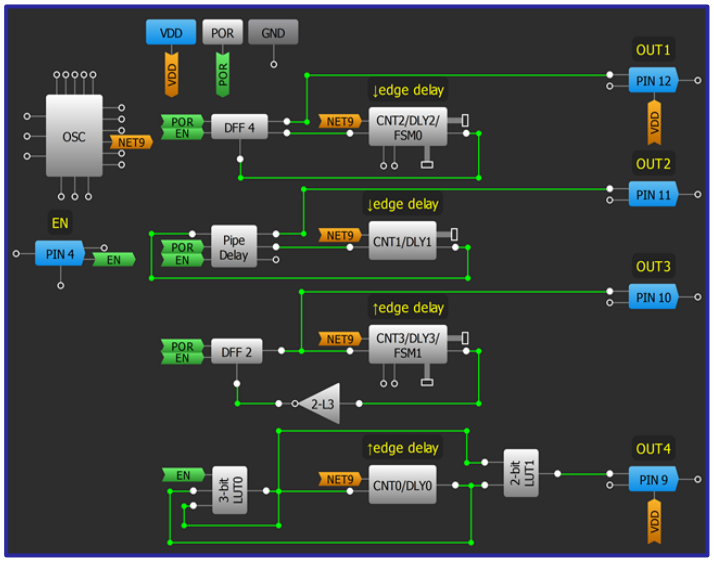


占空比 = 50% 的仿真结果

技巧：输出一个高脉冲信号

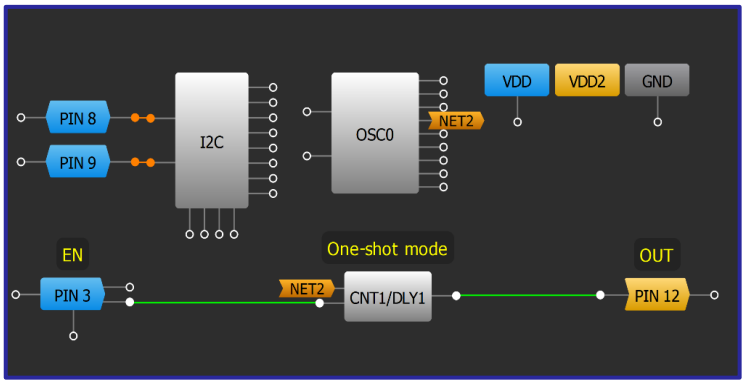
此技巧适用于所有 GreenPAK 芯片。

单脉冲电路生成具有事先定义的持续时间长度的输出脉冲。在电路产生脉冲后，它会返回其稳定状态，并且在再次触发之前不再产生脉冲。它是复位功能、看门狗定时器和许多其他应用中一个非常重要的组件。在 GreenPAK 中可以轻松输出一个脉冲信号。下图显示了几种由输入 EN 上升沿触发建立一次性脉冲的方法。脉冲持续时间可以通过改变所使用的 DLY 模块中的计数器数据值来调整。



几种一次性脉冲信号实现方法

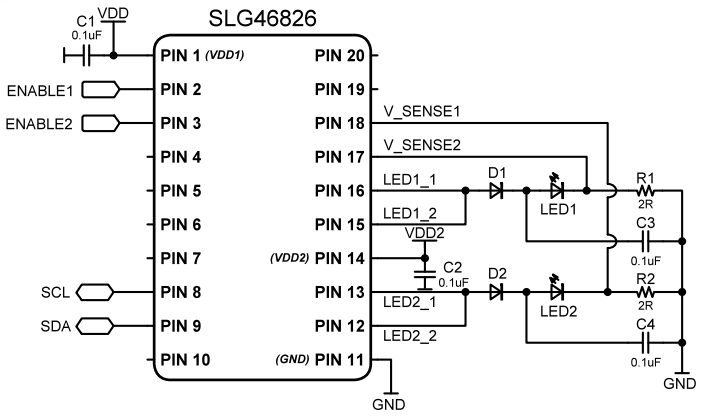
在许多 GreenPAK 芯片（见下图）中，实现一次性脉冲只需要一个 DLY 模块。用户唯一需要做的就是将模块属性窗口中的模式切换为“One shot”并选择它将检测到的边沿。可以是上升沿、下降沿或两者兼而有之。



使用 SLG46826 *实现一次性脉冲*

应用：LED 恒流驱动器

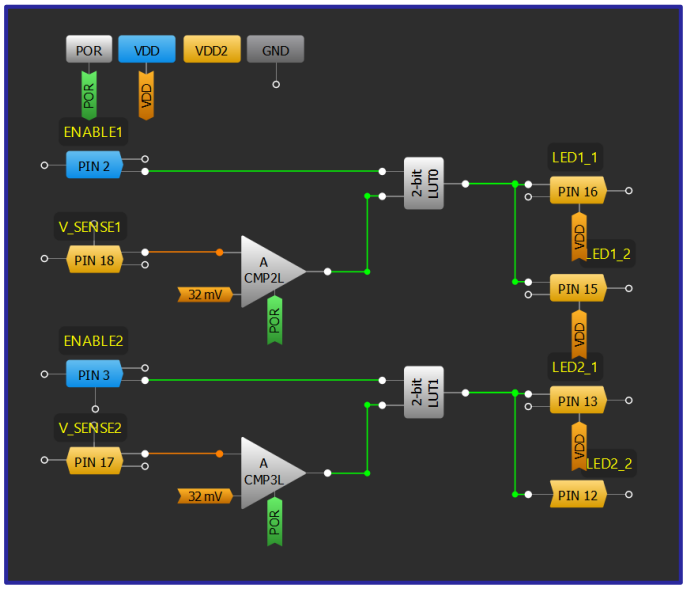
LED 驱动器为LED 提供恒定的电流，可以将输出功率维持在安全水平内，防止热失控。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 2 个电容
- 4 个二极管 (2 个 LED, 2 个普通二极管)
- 2 个电阻

GreenPAK 设计图

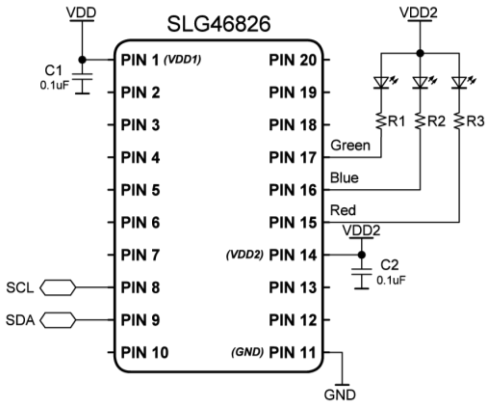


设计步骤

1. 配置 2 个 ACMP，并将每一个比较器的 **IN- source** 设置为所需的阈值。
2. 配置 LUT 逻辑以使能 LED 的输出。
3. 将引脚 **LED1_1** 和 **LED1_2** 连接到普通二极管的阳极，然后将普通二极管的阴极接到 LED 的阳极。
4. 在 LED 的阴极和 GND 之间插入一个电阻。
5. 在 LED 的阳极和 GND 之间并入一个电容。
6. 重复步骤 3 到 5 连接 LED2 的输出。
7. 将 **V_SENSE1** 脚和 **V_SENSE2** 脚分别连接到 LED1 和 LED2 的阴极。

应用：用 I²C 控制 RGB LED

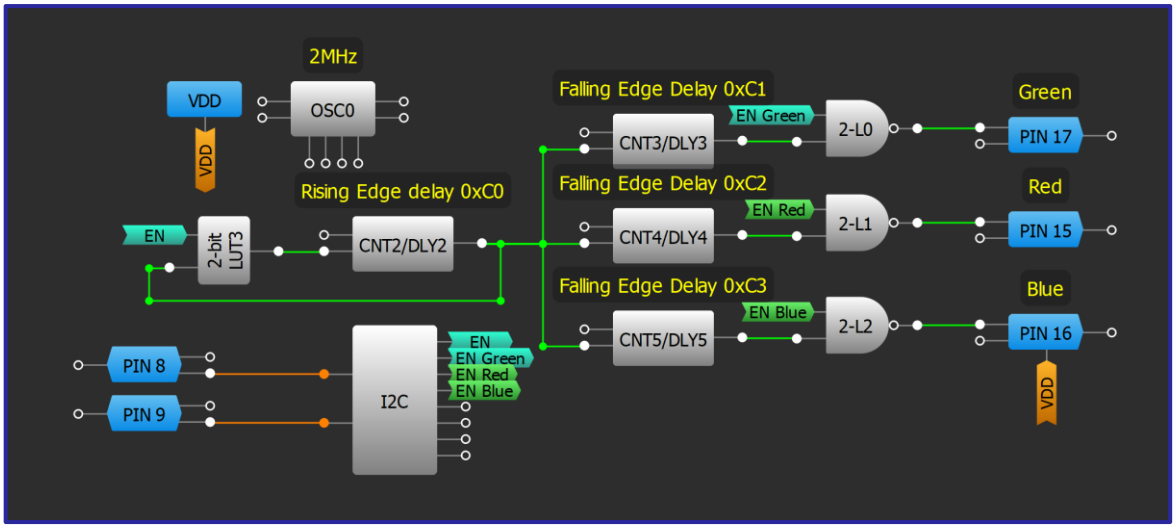
RGB LED 可用于多样化色彩输出 LED 显示系统，并可通过 GreenPAK 进行控制。在本实例中展示了一种利用 GreenPAK I²C，通过改变占空比以实现 LED 显示不同颜色的简单方法。



所需元器件

- 任意包含 I²C 的 GreenPAK IC(CMIC)
- RGB LED
- 3 个电阻

GreenPAK 设计图



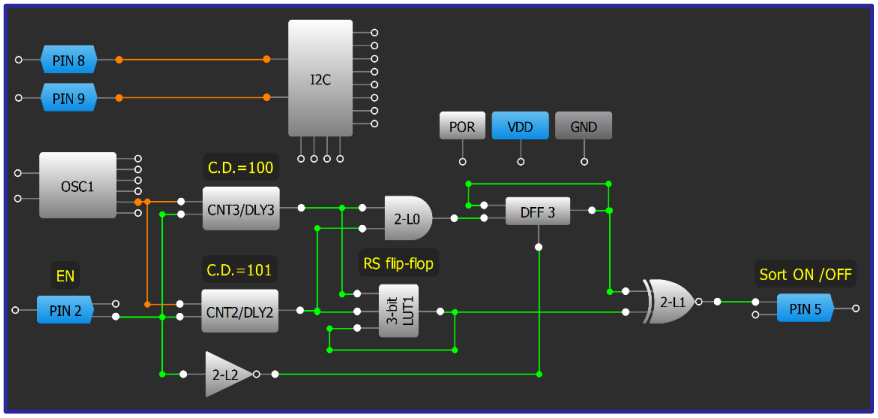
设计步骤

1. 将 RGB 阴极连接的 Pin 脚配置为 open-drain outputs。
2. 添加 LUT 逻辑单元和 **CNT/DLY2** 模块以创建一个包含 **EN** 使能信号的发生器。
3. 将 CNT/DLY 模块配置为 rising-edge delay。
4. 使用 技巧：配置标准逻辑 为各个输出添加并配置 LUT。
5. 将各个 LUT 的输出连接至相应的输出脚。
6. 通过访问 I²C Virtual inputs 对应的寄存器，每个 I²C Virtual Input 都可以单独更改或者同时更改。
7. 通过 I²C，可以单独或同时更改 CNT/DLY 模块的 Counter data。

技巧：LED 呼吸模式

此技巧适用于所有 GreenPAK 。 独立软开关通道的数量取决于特定芯片中可用的计数器数量。

呼吸 LED 模式可以利用两个计数器之间的恒定变化实现。在其编程周期内，每个计数器输出一个周期高电平 。 使用不同的 counter data settings 将两个 CNT/DLY 模块配置成计数值不同的计数器，让它们的输出之间产生一个小的偏移量。这些输出信号用于设置和复位设备内的触发器。 下图展示了一个基本实例，其中 **CNT2/DLY2** 设置 ON period, **CNT3/DLY3** 设置占空比。

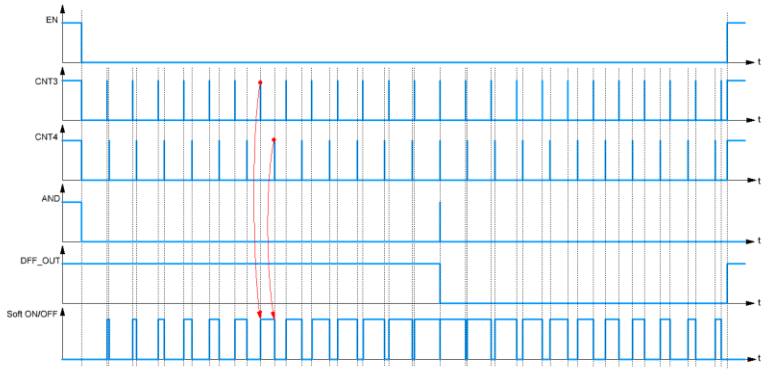


LED 呼吸模式实例

在上图的实例中，由 **CNT2** 所确定的 PWM 的频率可由以下等式计算：

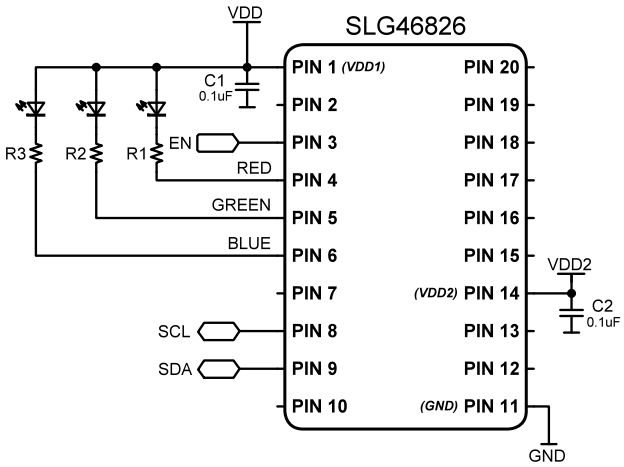
$$f_{PWM} = \frac{f_{OSC}}{(Data_{CNT2}+1)}$$

下图的波形显示了偏移量的影响。 当两个计数器的输出重合时，PWM 的周期结束。 这将导致在与门和 DFF 中出现一个很短的高电平脉冲。 或非门实现了 PWM 的反相，提供了一个软关断。 2 脚是使能信号，当其为高电平时，计数器将进行高电平复位。



应用：RGB LED 呼吸灯

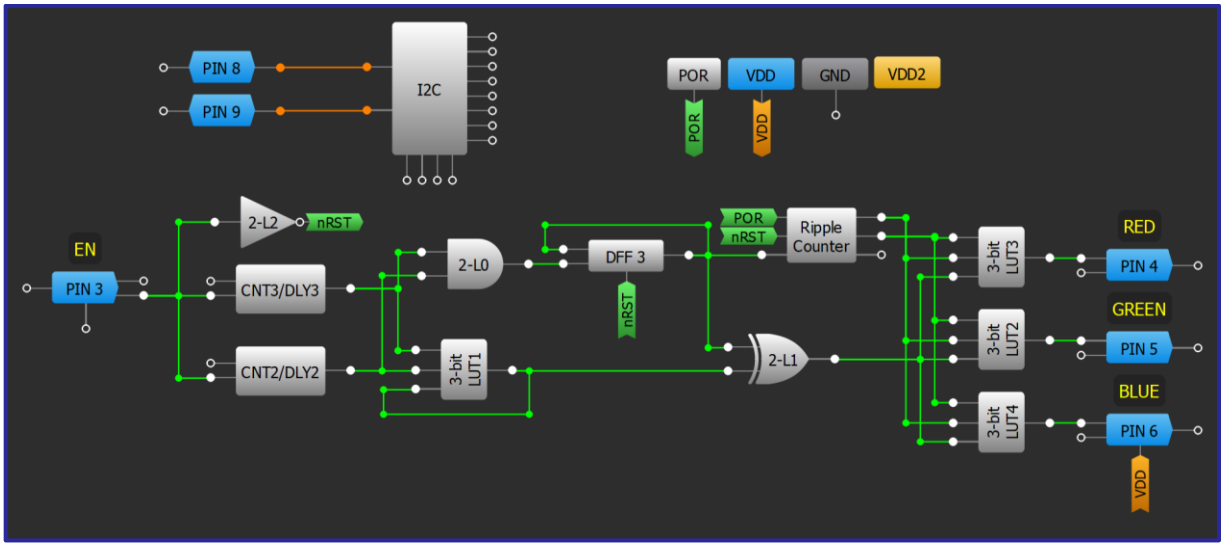
RGB LED 可用于多样化色彩输出LED 显示系统，并可通过 GreenPAK 进行控制。也可以搭配一个软开关电路来实现呼吸模式。



所需元器件

- 任意 GreenPAK IC(CMIC)
- RGB LED
- 3 个电阻

GreenPAK 设计图

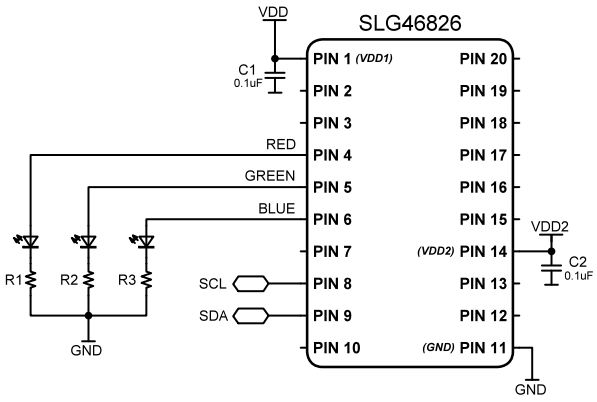


设计步骤

1. 将 GPIO 配置为 open drain NMOS。
2. 参考 技巧： LED 呼吸模式 创建一个软开关电路。
3. 配置 Ripple Counter 的 Functionality mode 为： range: SV-EV cycles (SV=1, EV=3)。
4. 将 LUTs 配置为一个多路解复用器。
5. 添加使能 （**EN**） 信号以控制 RGB 呼吸灯的启动/停止。

应用：用 I²C 实现 RGB LED 呼吸灯的控制

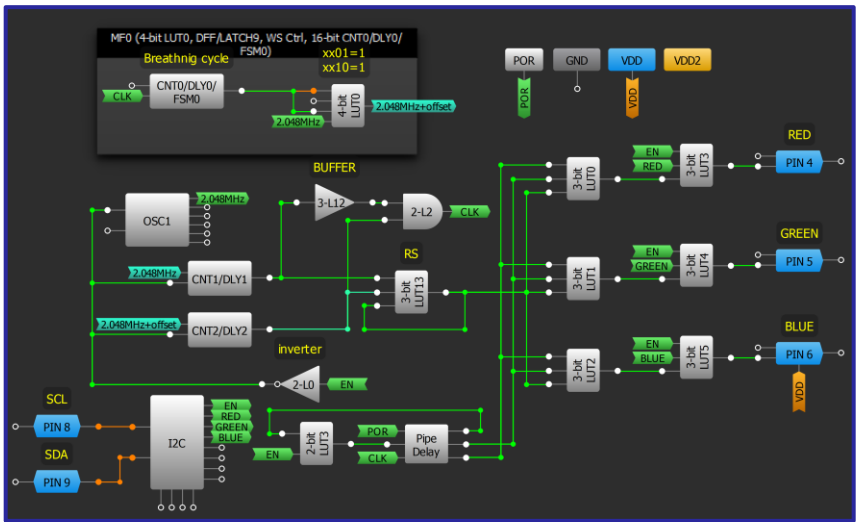
RGB LED 可用于多样化色彩输出LED 显示系统。也可以搭配一个软开关电路来实现呼吸模式，并可通过 GreenPAK I²C 进行控制。改变CNT0 的counter data 可改变呼吸周期。



所需元器件

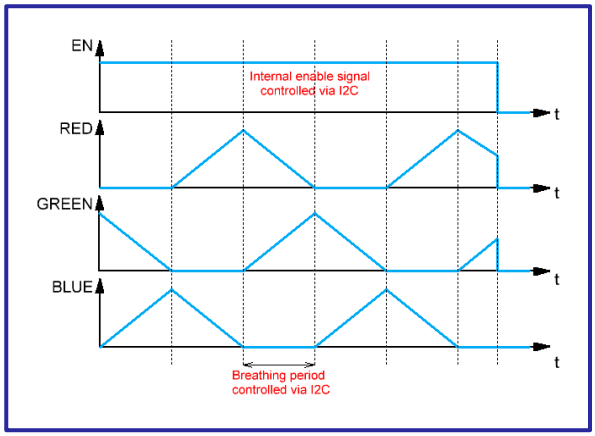
- 任意包含 I²C 的 GreenPAK IC(CMIC)
- RGB LED
- 3 个电阻

GreenPAK 设计图



设计步骤

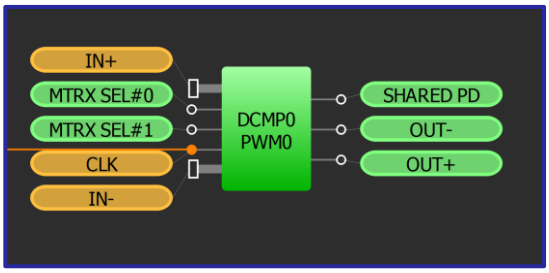
1. 参考 技巧： LED 呼吸模式 创建一个软开关电路， 需要将 **CN1** 和 **CN2** 的 counter data 设置为相同的值。
2. 添加 MF0 以建立 **CN1** 和 **CN2** 之间小的偏移量。
3. 将 **LUT3-LUT5** 配置为多路复用器， 由 I²C 的 EN 信号进行切换控制。
4. 将 **LUT0-LUT2** 配置为一个多路解复用器，它根据右侧的时序图传递呼吸信号。



技巧：DCMP/PWM 宏单元在 PWM 模式下的应用

此技巧适用于DCMP 组件，可应用于SLG46140, SLG46620 和SLG46621。

DCMP/PWM 宏单元概述



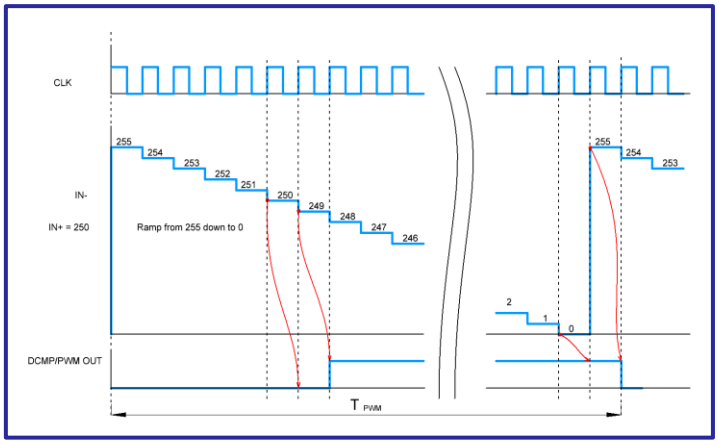
DCMP/PWM 宏单元可用于两个 8 位数值的比较或生成 PWM 信号。 每个 GreenPAK 芯片包含 3 个相互独立的 DCMP/PWM 组件，每个 DCMP/PWM 组件有两个 8 位输入端(IN+, IN-)，可以生成一个 PWM 信号。 输入脚 **MTRX SEL#0** 和 **MTRX SEL#1** 用于在静态 PWM 生 成过程中在 4 个寄存器中进行切换。 输入脚 **SHARED PD** 用于控制 DCMP/PWM 组件的启用或停用。 PWM 输出占空比范围可配置为 0% 至 99.61% 或 0.39% 至 100% 。

创建 PWM 信号

PWM 发生器的一个输入信号来自于计数器，它能根据 PWM ramp 值线性循环，计数范围从 255 到 0，反之亦然。

另外一个输入至少在 1 个 PWM 信号周期内 (PWM 斜坡计数器周期) 是稳定的。它可以是来自 SPI、ADC、FSM 组件的数据，也可以是来自 DCMP/PWM 的内部寄存器的数据。

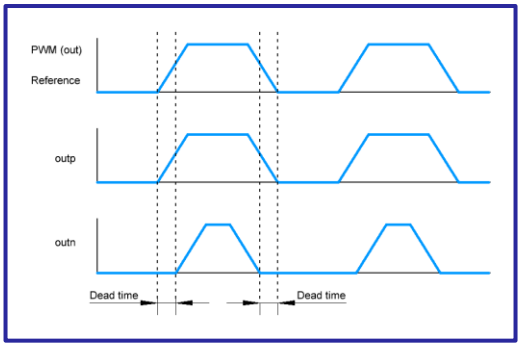
右图显示了当 **IN-** 连接到一个由 255 计数到 0 (PWM 斜坡计数器)的 CNT/DLY 组件，**IN+** 是一个设置为 250 的内部寄存器时 DCMP/PWM 的工作波形。



DCMP/PWM 在 PWM 模式下的时序

IN+ 信号的配置是该宏单元配置的关键因素。可以通过内部寄存器设定静态 PWM 值，通过 ADC 监控 PWM 动态反馈，MCU 也可以通过 SPI 接口调节发出的 PWM。

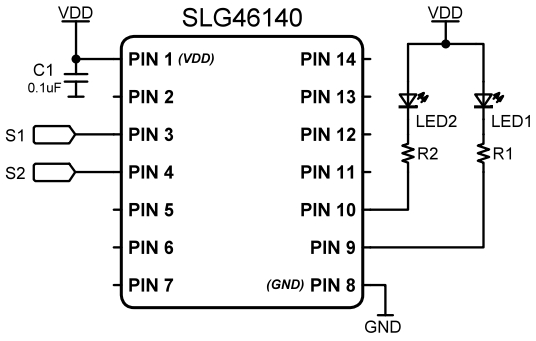
输出 **OUT-** 和 **OUT+** 的死区时间可以设定为 10 - 80 ns, 可以通过 DCMP/PWM 的属性面板进行设定。



OUT- 和 OUT+ 的死区时间

应用：PWM 选择器

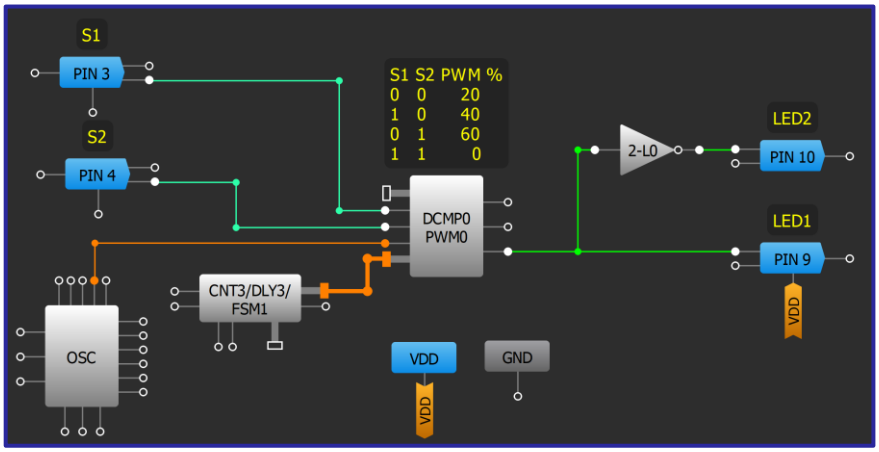
PWM 选择通常用于 LED 亮度调节和风扇转速控制等功能。在本实例中，基于两个按键的输入，可实现两颗LED 的亮度调节。



所需元器件

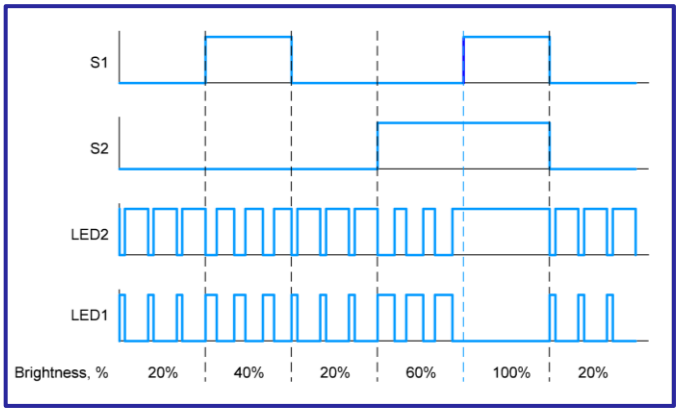
- 任意包含 DCMP 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



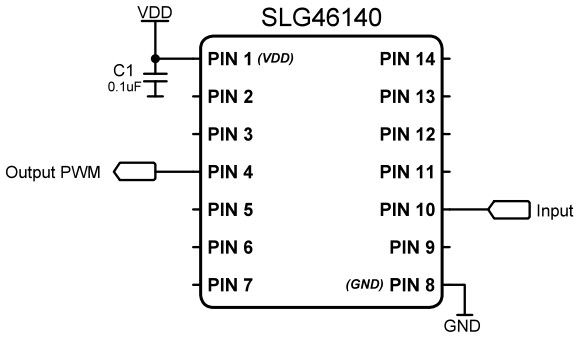
设计步骤

1. 断开 VDD 与 **SHARED PD** 的连接，启用 DCMP。 将 DCMP/PWM 的 power register 设置为 **Power on**, 将 **IN+ selector** 配置为 **Register selected through from matrix**, **IN- selector** 配置为 **FSM1[7:0]**。 将 DCMP0 的各个寄存器配置如下： register 0 – 51; register 1 – 102; register 2 – 154; register 3 – 0。
2. 添加 CNT/DLY 模块并配置 Counter/FSM， 设置 counter data 为 255。
3. 将 RC OSC power mode 配置为 Force power on 。
4. 将 **PIN10** 和 **PIN9** 配置为 open drain NMOS。
5. 添加 LUT 作为反相器。
6. 将输入脚连接到 DCMP/PWM 模块的 **MTRX SEL** 脚。



应用：采用 ACMP 和 DAC 的 PWM 发生器

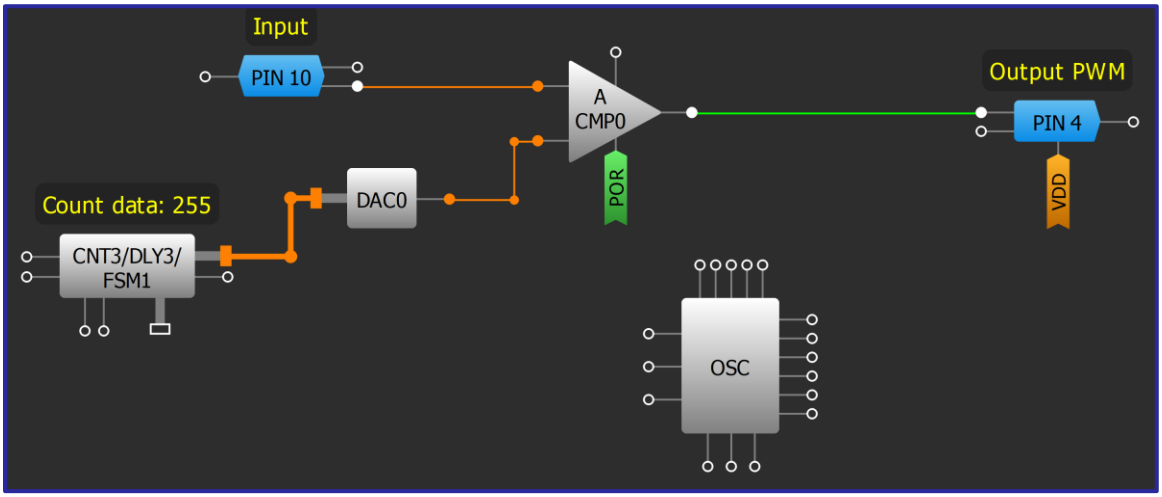
PWM 发生器可以用来控制直流电机和 LED 等设备。本实例使用 ACMP 将输入的模拟信号和 DAC0 输出信号进行比较，CNT3 用于生成 DAC 的值。



所需元器件

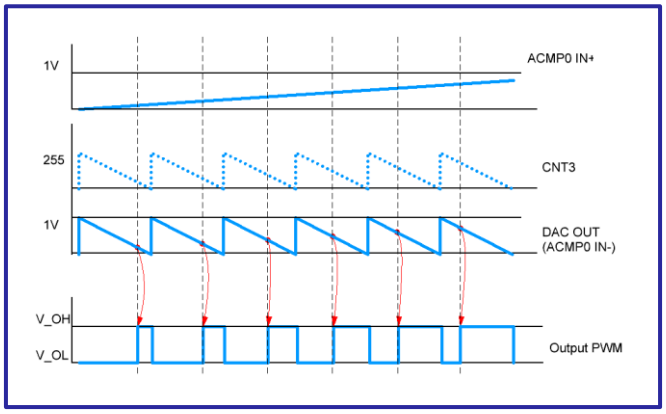
- 任意包含 DAC 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



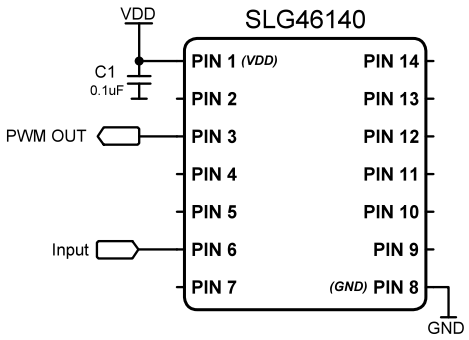
设计步骤

1. 将 ACMP0 的 **PWR UP** 连接到 POR，**IN- source** 配置为 **Ext. Vref (DAC0 out)**。
2. 将 **DAC0** 的 power on signal 配置为 **Power on**， **input selection** 选择为 **From DCMP1's input**。
3. 将兼容 FSM 的 CNT/DLY 配置为 **Counter/FSM** 模式，设置 counter data = 255。
4. 将 RC OSC power mode 配置为 **Force power on**。



应用：采用 ADC 的 PWM 发生器

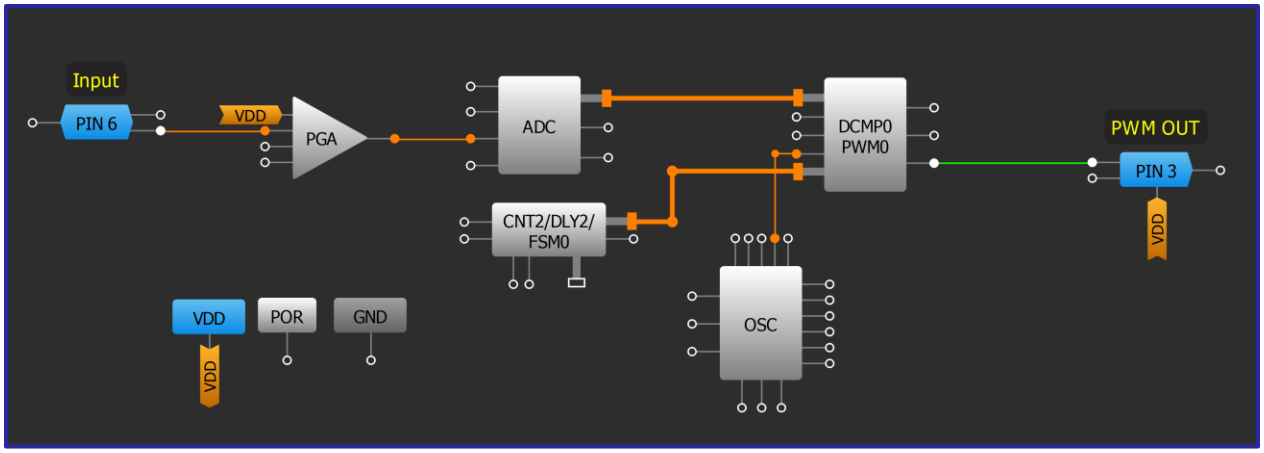
PWM 发生器可以用来控制直流电机和 LED 等设备。本实例使用连接到 ADC 的模拟信号与PWM0 中的CNT2 值进行比较。如果CNT2 值小于被数字化的模拟信号值，PWM0 输出高电平。CNT2 值变为0 后，PWM0 输出低电平。



所需元器件

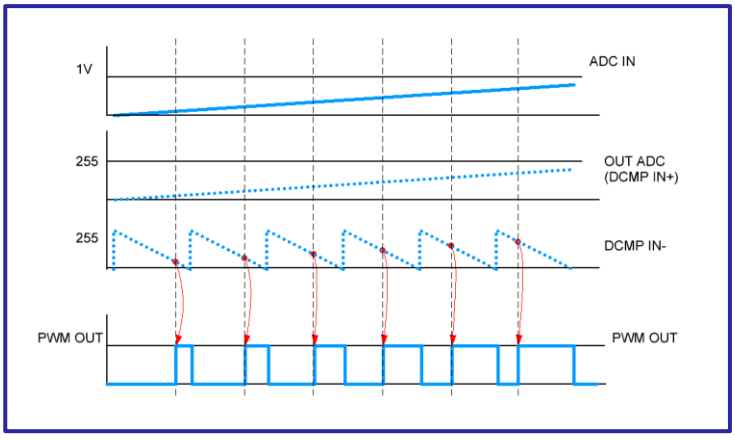
- 任意包含 ADC 的 GreenPAK IC(CMIC)
- 无需其他元器件

GreenPAK 设计图



设计步骤

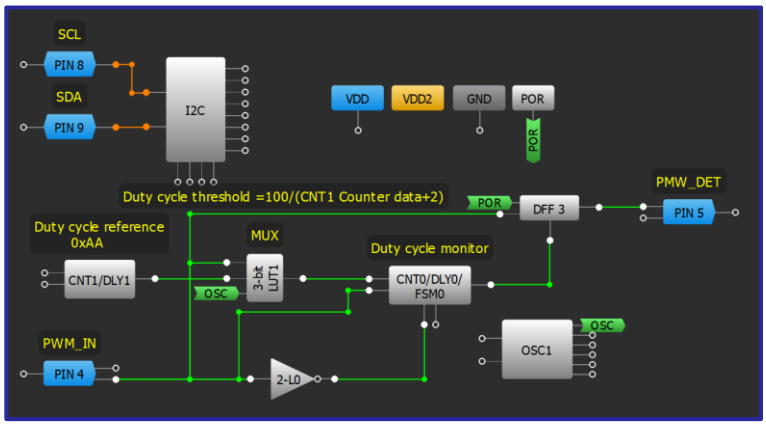
1. 断开 VDD 与 ADC-**PWR DOWN** 的连接，将 PGA **power on signal** 配置为 **Power on**.
2. 将 **PIN 6** 配置为 **Analog input/output**。
3. 配置 **DCMP0/PWM0**， 断开 VDD 与 **SHARED PD** 的连接。DCMP/PWM power register 配置为 Power on。确认 IN+ selector 连接到 **ADC [7:0]** ， **IN- selector** 连接到 **FSM0 [7:0]**。
4. 配置一个 **4 位的 LUT1/ 14 位的 CNT2/DLY2/FSM0** 作为 **Counter/FSM**， 设置 counter data 为 255 。
5. 将 **DCMP0/PWM0 OUT+** 连接到输出脚。



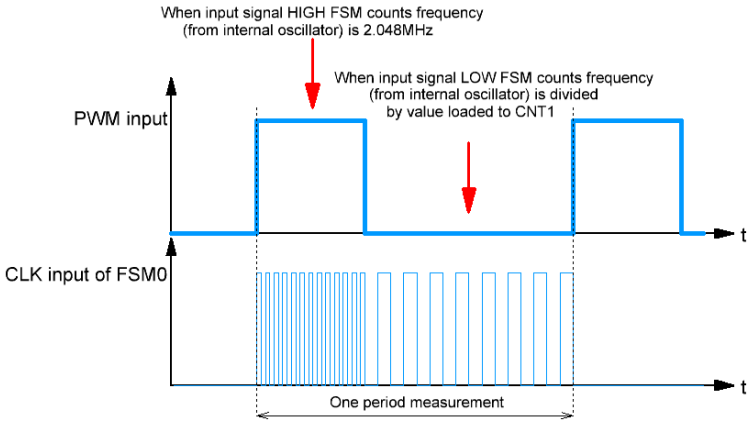
技巧：占空比检测

此技巧适用于所有 GreenPAK 。 由于输入频率范围受到最大 FSM 计数器数据的限制，所以最好使用 16 位 FSM。PWM 检测输入频率应大大低于占空比参考频率，以提高精度。

占空比检测对于过载保护、DC/DC 转换、伺服电机控制和协议检测等功能非常重要。使用包含 FSM 组件的 GreenPAK 可以很容易地实现这种设计(以下实例采用了 SLG46826，如下图)。



在上述实例中, 当 **4 脚**转为高电平，**FSM0** 根据内部振荡器时钟频率，开始向下计数(减小计数值)。 在 **4 脚**脉冲的上升沿，**FSM0** 被置为 65535。 当 **4 脚**转为低电平，**FSM0** 开始以内部振荡器除以 **CNT1** 的值为频率进行向上计数。 如果 **FSM0** 计数值达到 65535，**DFF3** 将在 4 脚的下一个上升边沿转为低电平，这表示占空比低于设置的阈值。

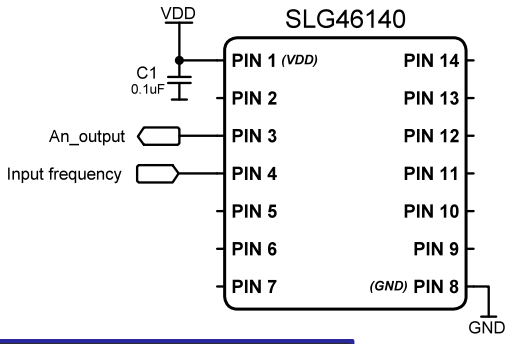


占空比的参考频率可通过利用 I²C 来改变 **CNT1** 的 counter data value 以进行调整。 使用以下公式计算占空比阈值：

$$Duty\ Cycle\ Threshold = \frac{100}{\text{CNT1 Counter data} + 2}\ (\%)$$

应用：频率/模拟电压转换器

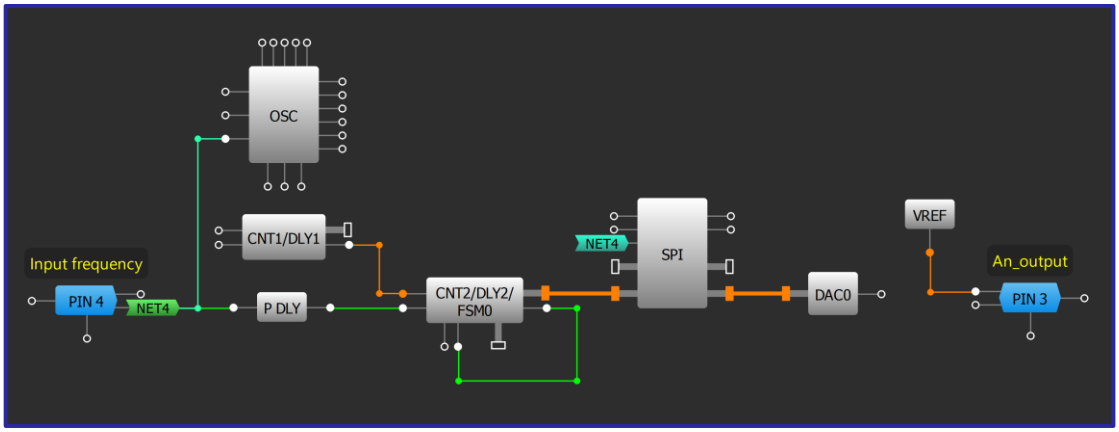
该应用将输入频率转换为模拟电压。 输入频率在一定范围内可由设计元件选择和调节。 输出模拟电压恒定并且可根据需要进行调整。

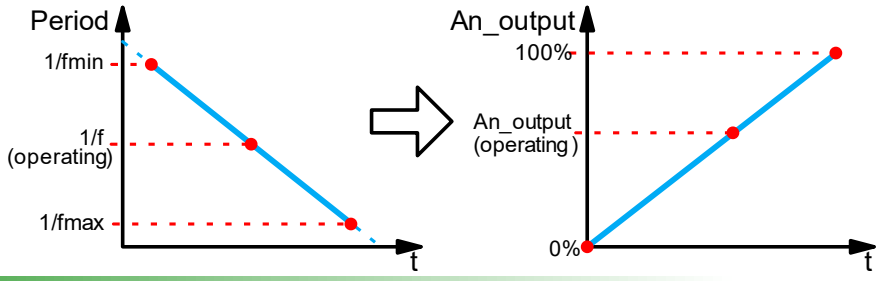


所需元器件

- 任意带 SPI 和 DAC 功能的 GreenPAK IC (CMIC)

GreenPAK 设计图





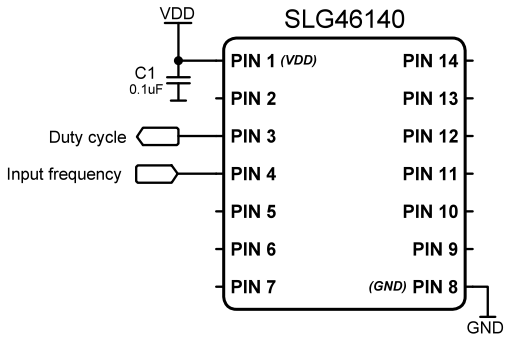
设计步骤

1. 配置 SPI 为“ADC/FSM Buffer”模式，修改 **PAR input data** 源为“FSM0[15:8]。
2. 配置 **FSM0** 模块为“Set (counter value)”，将时钟源修改为 **CNT1**。
3. 配置 DAC **Input selector** 为 “From DCMP1 's Input”，VREF **Source selector** 为 “DAC0 out”。
4. 输入频率范围和输出模拟电压的计算公式如下:

$$f_{min} = \frac{fosc}{(CNT1+1)\cdot FSM0} \qquad f_{max} = \frac{fosc}{(CNT1+1)} \qquad An\_output_{operating} = V_{ref\_max} - \frac{f_{min}\cdot V_{ref\_max}}{f_{operating}}$$

应用：频率/占空比转换器

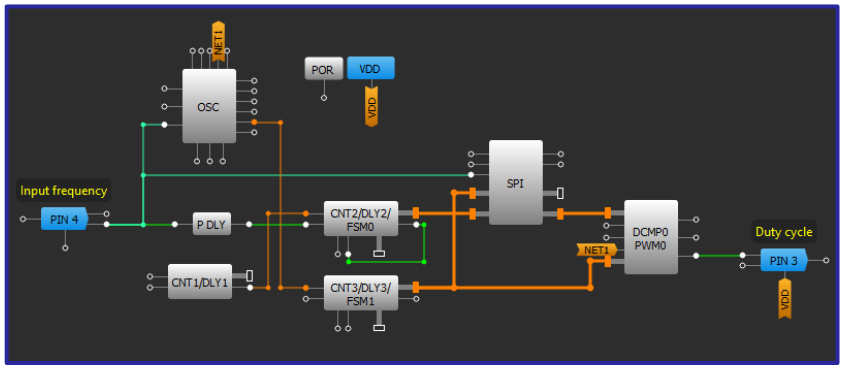
该应用可以用来将输入频率转换为一定的占空比。 输入频率在预定的范围内，可以通过调整计数器来改变。 输出 PWM 频率是恒定的，但可以根据给定的要求改变。

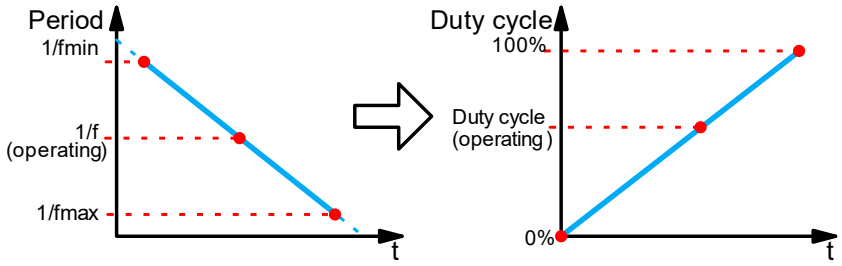


所需元器件

- 任意带 SPI 跟 DCMP 的 GreenPAK IC (CMIC)

GreenPAK 设计图





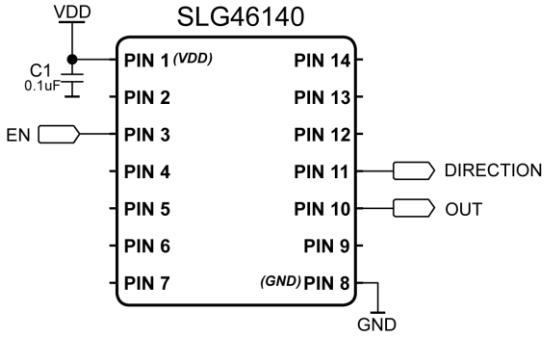
设计步骤

1. 配置 SPI 为“ADC/FSM buffer”模式，修改 PAR 输入数据源为“FSM0[15:8] FSM1[7:0]”。
2. 配置 **FSM0** 模块为“Set (counter value)”，将时钟源修改为“8 位 CNT1/DLY1 (OUT)”。
3. 配置 DCMP0 为比较“SPI[15:8]”数据和“FSM1[7:0]”数据。
4. 使用以下公式计算输入频率范围和工作占空比：

$$f_{min} = \frac{fosc}{(CNT1+1)\cdot FSM1} \qquad f_{max} = \frac{fosc}{(CNT1+1)} \qquad Duty\ Cycle_{operating} = \left(1 - \frac{f_{min}}{f_{operating}}\right)\cdot 100\%.$$

应用：线性调频

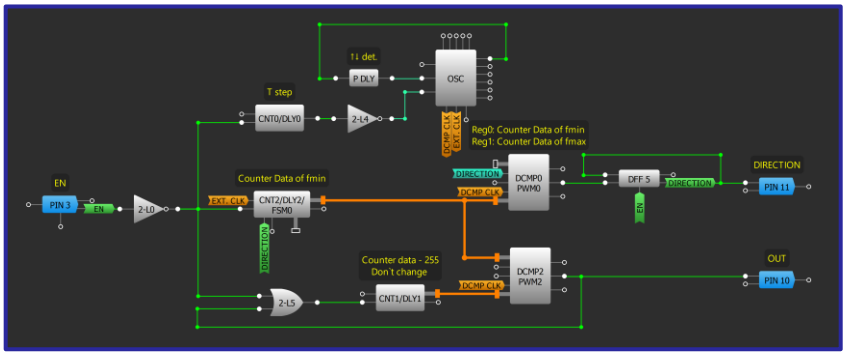
该应用可以用来产生一个频率，它的频率可以在特定的时间内从 fmin 逐渐改变为 fmax，反之亦然。在较大的频率范围内，频率不会线性变化。

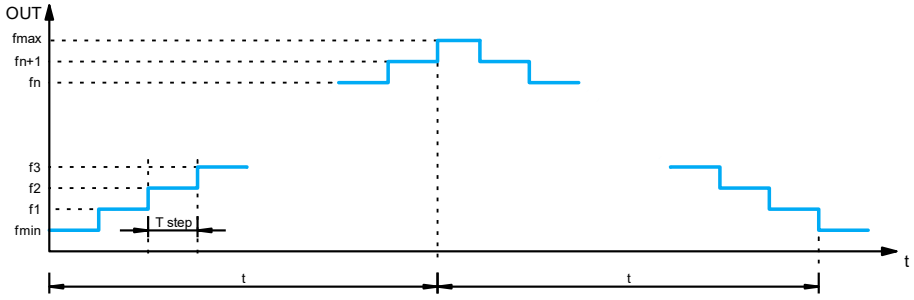


所需元器件

- 任意带 DCMP GreenPAK IC (CMIC)

GreenPAK 设计图





设计步骤

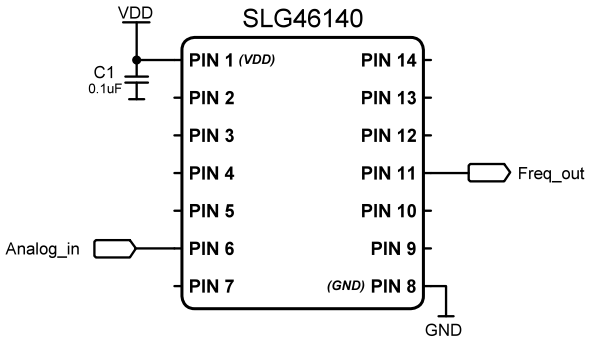
1. 配置所需的 GPIO 引脚。
2. 如上图所示，添加连接并配置 LUTs、P DLY、DFF、DCMPs 和 CNT/DLY/FSM 模块。
3. 最小、最大频率计数器数据及上升/下降周期计算如下：

$$Counter\ Data_{\max\,(min)} = 255 - \left(\frac{f_{osc}}{f_{\max\,(min)}} - 1\right) \qquad T_{step} = \frac{t \cdot f_{max} \cdot f_{min}}{f_{osc} \cdot (f_{max} - f_{min})}$$

4. 为特定的 DCMPs 选择器设置适当的 counter data。
5. 为步长时间 T_{step} 设置适当的 **CNT/DLY0** counter data。

应用：压控振荡器

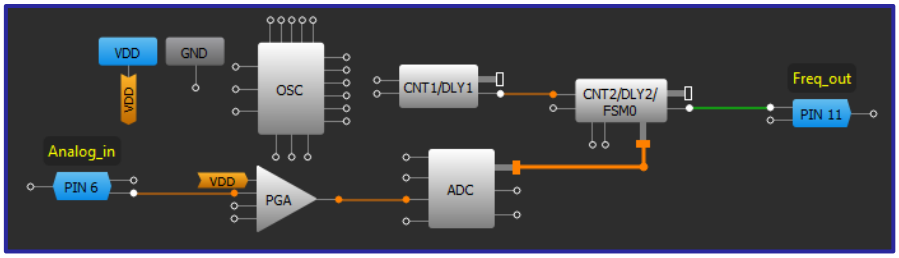
本设计可用于将输入模拟电压转换为频率。该频率由 FSM0 生成，其 counter data 由 ADC 确定。FSM0 的 counter data 决定了分频系数。为了增加频率范围，CNT1 被用作预分频器。

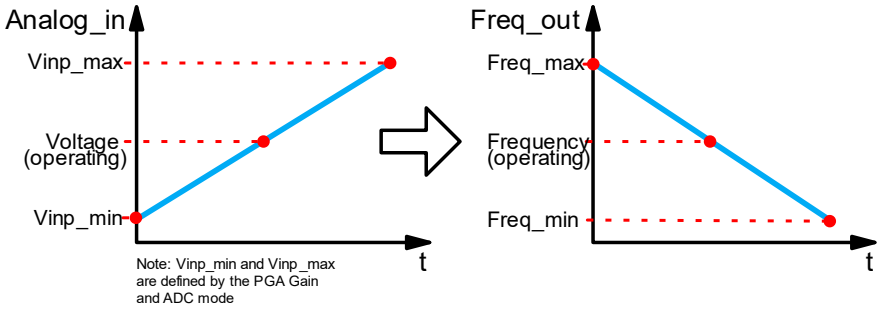


所需元器件

- 任意带有 ADC 的 GreenPAK IC (CMIC)

GreenPAK 设计图





设计步骤

1. 将 PIN6 配置为 Analog input/output。
2. 修改 **FSM0** 的连接：将 **FSM data source** 连接到 ADC，**Clock** 连接到 8 位的 CNT1/DLY1(输出脚)。
3. 将 ADC 的 Power on signal 配置为 Power up, 将 PGA **Gain** 由“x0.25” 修改为 “x1”。
4. 利用以下公式计算输出频率范围和工作频率：

$$fmax = \frac{fosc \cdot Vinp\_max}{2 \cdot (CNT1+1)} \qquad fmin = \frac{fosc}{256 \cdot (CNT1+1)} \qquad Frequency(operating) = fmax \cdot \frac{Vinp\_max}{Voltage(operating)}$$

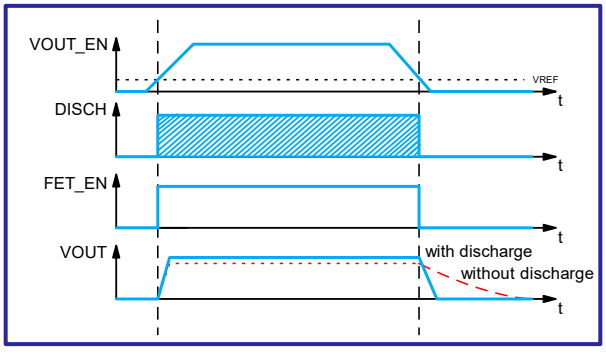
第 7 章:电源管理

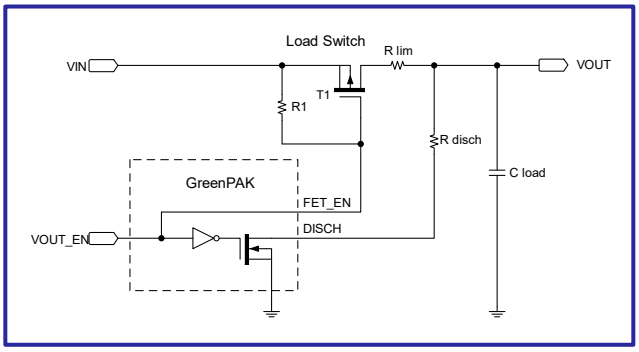
本章介绍电子系统中电源管理的相关应用。一些涉及到电源管理的应用有电荷泵、LDO 及放电电路。

技巧：输出放电

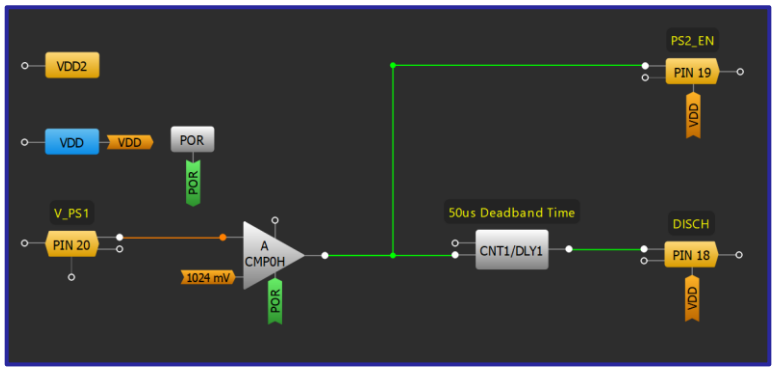
此技巧适用于所有 GreenPAK。

输出放电电路是一种防止输出引脚悬空或系统欠压的方法。当器件未启用但其在工作期间没有泄放回路时，输出放电电路可以确保输出引脚设置为已知的“零”状态。





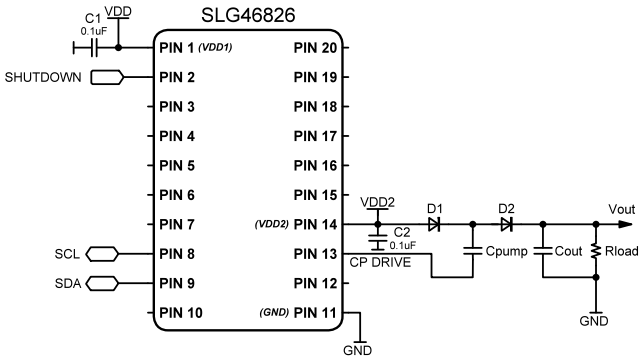
上图显示了负载开关输出 **VOUT** 的快速放电电路的实现方法。GreenPAK 中的开漏 NMOS 输出的漏极连接到 **VOUT**。 NMOS 的栅极在 GPIO 结构内是反相的，因此不需要在矩阵内反相 **VOUT_EN** 信号。当 **VOUT_EN** 为高电平时，负载开关打开，到 GND 的放电路径断开以防止泄放。当 **VOUT_EN** 为低电平时，负载开关关闭，到 GND 的放电路径闭合以快速对 VOUT 放电。为了限制通过 FET 的电流并控制放电，添加了两个电阻器 Rlim 和 Rdisch。



上图显示了电源时序器中的条件输出放电电路示例，该电路监控其之前的电源电平。将 **PS2_EN** 配置为推挽输出，将 **DISCH** 配置为开漏 NMOS 输出。当 **V_PS1** 降至 1024mV 以下时，该电路将关闭电源并将输出对 GND 放电。**CNT/DLY1** 增加了一个 50µS 的死区时间。

应用：电荷泵

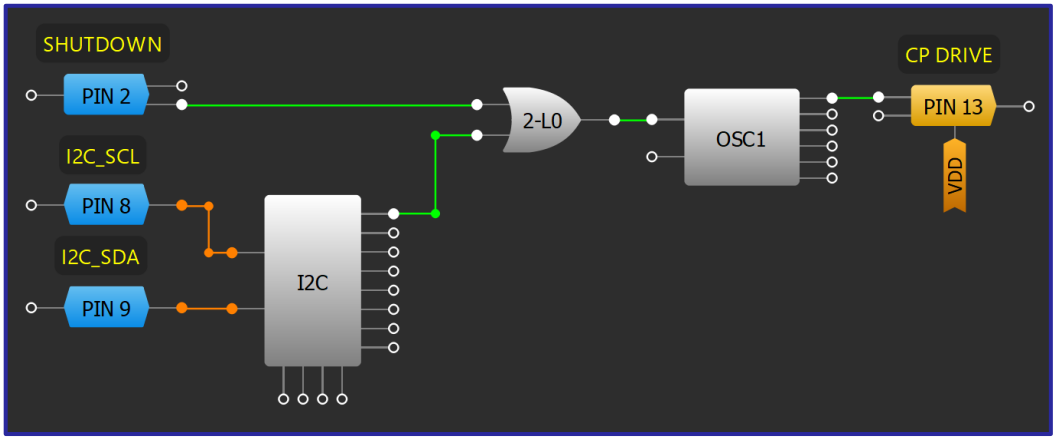
电荷泵是一种利用电容进行能量存储的 DC-DC 转换器，以产生高于输入的电压，应用于传感器或者特定的接口电路以提供更高的电压。为了达到最佳性能，推荐使用肖特基二极管。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 2 个电容
- 2 个二极管

GreenPAK 设计图

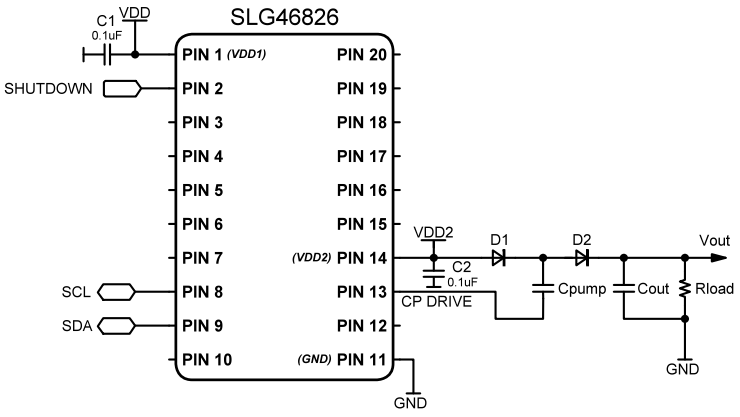


设计步骤

1. 在 **OSC1** 中设置分频器以获得所需的输出频率。
2. 配置逻辑以提供关闭充电泵的接口[I/O 和 I²C]。
3. 在 VDD 和 **CP_DRIVE** 之间连入二极管 D1 和电容 C_{pump}。
4. 将 D1 的阴极与 D2 的阳极相连。
5. 将 C_{out} 和 R_{load} 并联在 D2 的阴极和 GND 之间。

应用：二级电荷泵

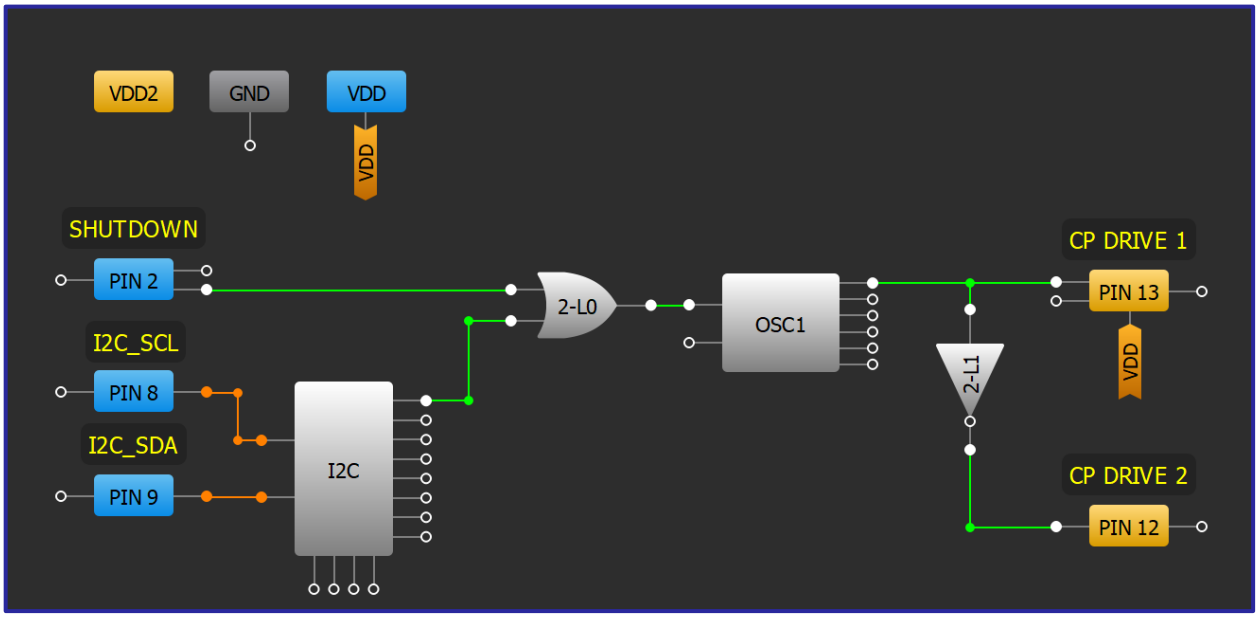
电荷泵是一种利用电容进行能量存储的 DC-DC 转换器，以产生高于输入的电压。多级电荷泵可以将输出电压推至输入电压的二倍以上，为了获得最佳的性能，推荐使用肖特基二极管。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 3 个电容
- 3 个二极管

GreenPAK 设计图

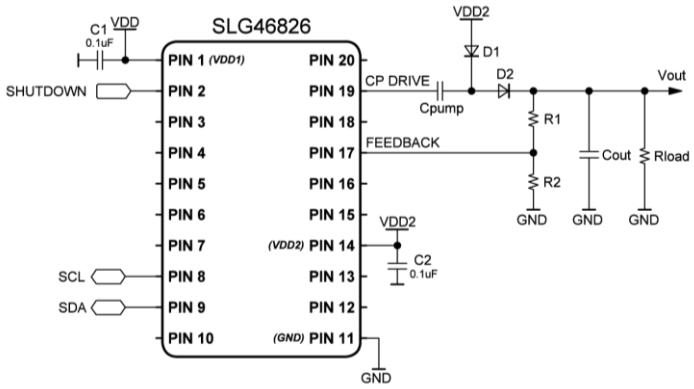


设计步骤

1. 根据 应用：电荷泵 的步骤创建基本电荷泵。
2. 添加一个 LUT 并配置为 Inverter，将这个反相器连接在振荡器输出和另一个电容驱动脚之间。
3. 在电荷泵的第一级和输出级之间添加额外的二极管和电容，新添加的二极管和电容应该和第一级使用的二极管和电容具有相同的类型和大小。

应用：具有输出调节功能的电荷泵

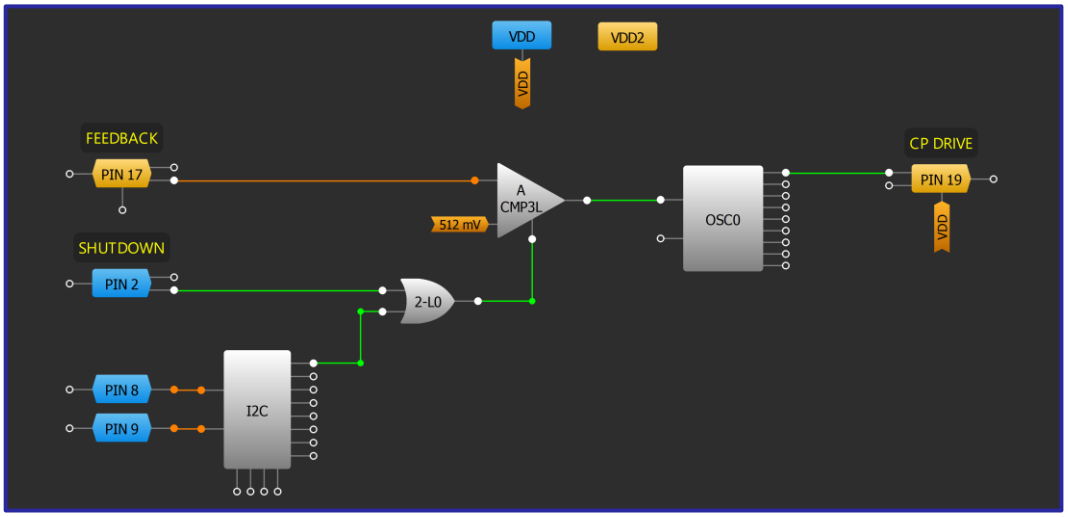
电荷泵是一种 DC-DC 转换器，它使用电容器进行高能电荷存储以产生不同的电压阈值等级。这种带有输出调节电路的电荷泵可以通过I²C 改变输出电压值。建议使用肖特基二极管以实现最佳性能。



所需元器件

- 任意带有 ACMP 的 GreenPAK IC (CMIC)
- 两颗电容
- 两个二极管
- 两只电阻

GreenPAK 设计图



设计步骤

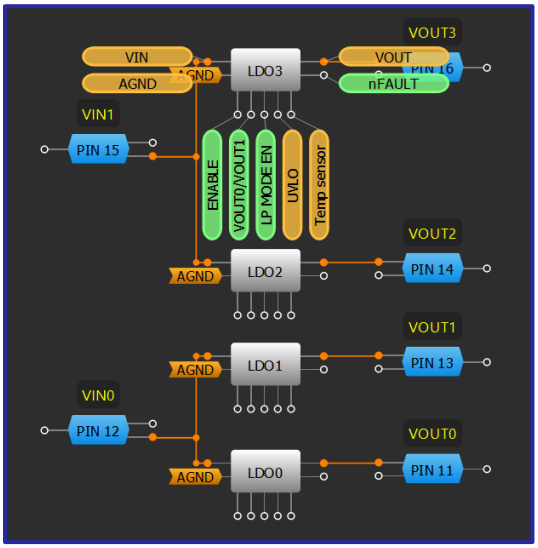
1. 参考 应用：电荷泵 创建一个基本的电荷泵设计。
2. 将 **ACMP3L** 的 **IN+ source** 配置为 **PIN17 (FEEDBACK)**，**IN- source** 设置为所需的基准电压。
3. 使用两只电阻 R1 和 R2 建立一个分压电路，将分压输出连接到 **PIN17**。
4. 利用公式 $V_{out} = V_{ref} \cdot \frac{R1}{R2}$ 计算出输出电压。
5. GreenPAK 可以通过 I²C 设置 **ACMP3L** 的 **IN- source** 基准电压，改变输出电压。也可以通过改变反馈电阻分压器系数，以将输出电压调节到不同的值。

技巧：LDO 稳压器

此技巧介绍了 SLG46580、SLG46582、SLG46583 和 SLG46585 中的低压差 (LDO) 稳压器宏单元。

尽管负载阻抗发生变化或电源电压发生变化，低压差 (LDO) 稳压器仍可将电源的输出电压维持在一个稳定值，并具有最小的压差电压。这使得它们对于使用干电池的便携式设备和必须处理周期性波纹的射频系统非常有用。一些 GreenPAK 芯片配备了 LDO 稳压器宏单元。

每个 LDO 宏单元有 32 种可能的输出电压等级，范围从 0.90V 到 4.35V。可以对两种不同的输出电压（VOUT1 和 VOUT2）进行编程。根据输入的状态确定 VOUT1/VOUT2 输出哪个电压电平。有关每个输出电压对应的最小 VIN 和 VDD 值，请参阅 SLG46580 数据表。

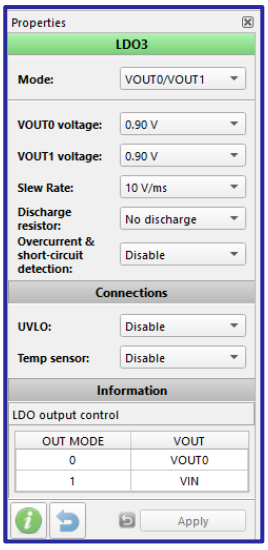


LP MODE EN 输入端子配置为 LOW（默认），选择高功率 (HP) 模式，将其配置为 HIGH，选择低功率 (LP) 模式。 HP 模式可以实现最高额定输出电流，但 LP 模式具有更小的静态电流，可在更小的额定值内提供更高的效率。只有在每个 LDO 的 VOUT 脚上都连接 1 个 >2µF 的电容时，LDO 才能在 HP 模式下保持稳定。

在属性面板中，可以更改每个 LDO 的 slew rate 以设置软启动。每个 LDO 都可以选择在其输出上启用 300 Ohm 放电电阻。每个 LDO 还可以同时启用 210mA 的过流限制和短路检测，如果在 HP 模式下输出电压降至 0.5V 以下，则将电流限制为 20mA。

在 LDO 中启用 UVLO 连接，可以设置 ACMP 检测其 VIN，并在 VIN 低于某个欠压锁定(UVLO)阈值(ACMP in - level)时关闭 LDO。

配置 LDO 稳压器时必须考虑器件的散热。这些器件在 85° C 环境温度下的额定功耗为 0.6W。启用 **Temp sensor**，将 ACMP2 配置为检测 GreenPAK 的温度传感器，以便在超过适当的编程温度时关闭 LDO，并在 LDO 冷却后重新启动。

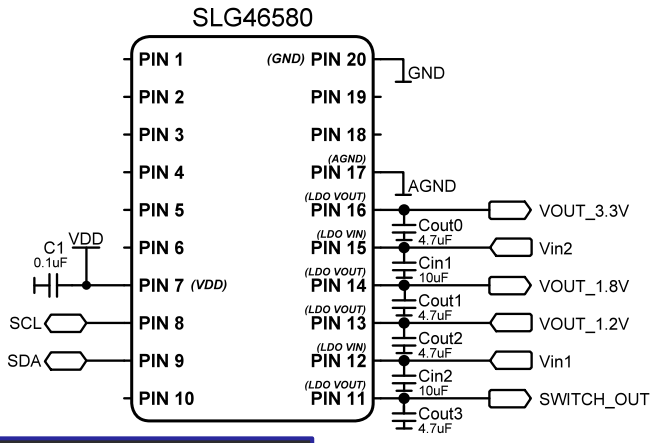


器件规格

- SLG46580/SLG46585:
- 4 个 LDO 稳压器;
- Imax: HP Mode = 150mA, LP Mode = 100µA
- SLG46582:
- 2 个 LDO 稳压器;
- HP Mode Imax = 300mA, LP Mode Imax = 200µA
- SLG46583:
- 1 个 LDO 稳压器
- HP Mode Imax = 600mA, LP Mode Imax = 400µA

应用：灵活的电源岛

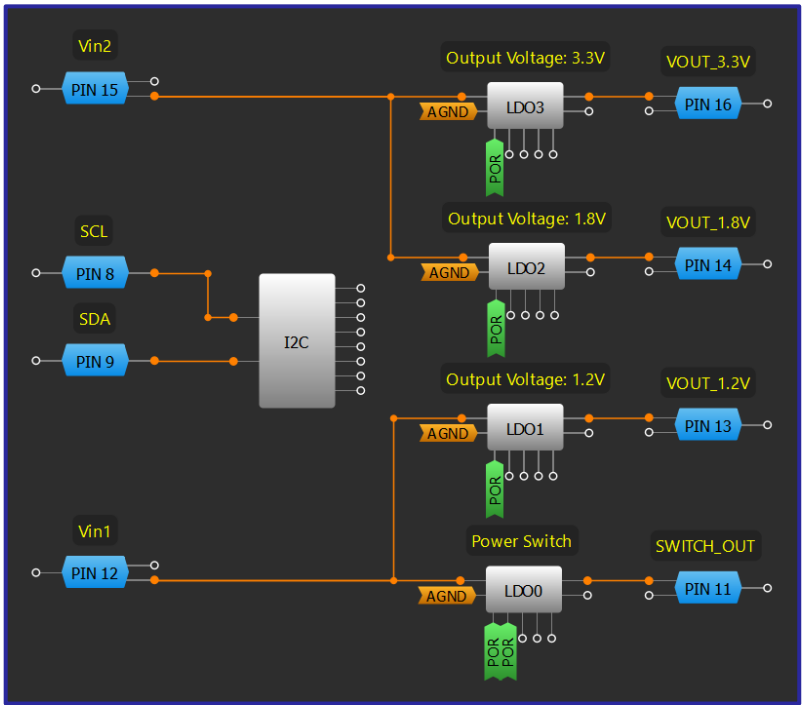
灵活的电源岛可以帮助设计人员将电源系统划分为分布在整个系统中的电源区域的小“岛”。它可以提供不同级别的调节电压以满足特定的系统要求。



所需元器件

- 任意带有内部 LDO 的 GreenPAK IC (CMIC)
- 7 颗电容

GreenPAK 设计图

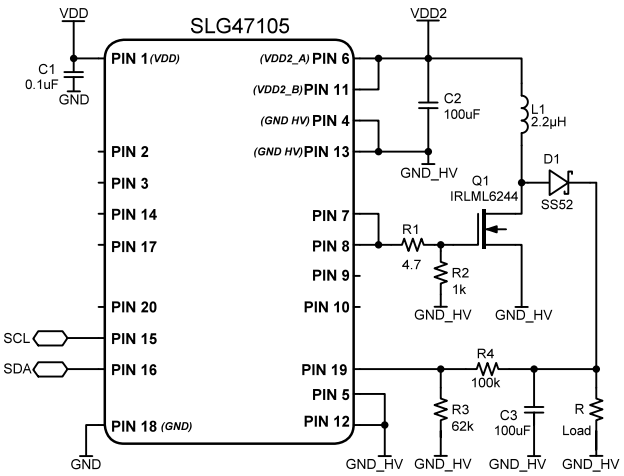


设计步骤

1. 将每个 **LDO** 的 **VOUT0 电压**和 **VOUT1 电压**设置为所需的电压。
2. 为每个 LDO 的 **ENABLE** 脚连接 **POR**。
3. 将一个 LDO 配置为 **VOUT0/PWR switch** 模式，并设置 **OUT MODE** 脚为高电平以将其设置为电源开关输出。

应用：升压转换器

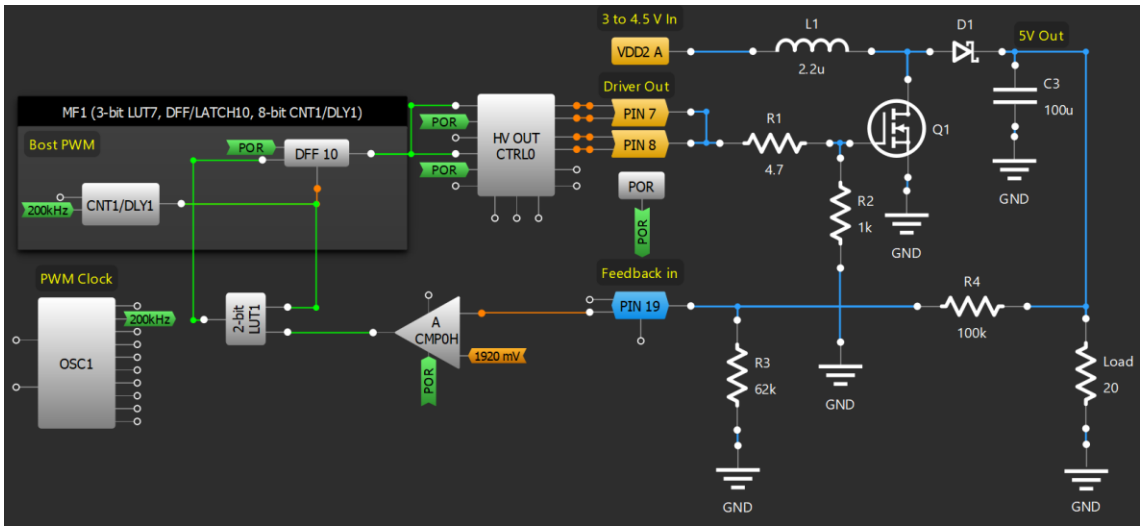
本实例展示了 SLG47105 作为升压转换驱动器的方法。在本设计实例中，升压转换器将 VDD2 电压从 3V 升高到 5V。



所需元器件

- SLG47105
- MOSFET
- 肖特基二极管
- 电感
- 3 个电容
- 4 个电阻器

GreenPAK 设计图

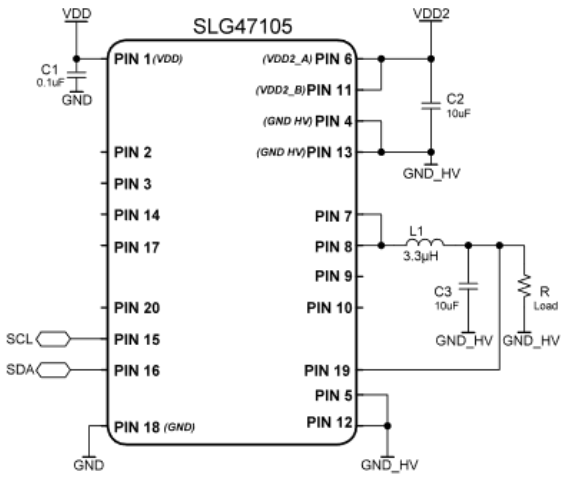


设计步骤

1. 完成电路设计。按照上述电路图所示连接所有组件。
2. 启用 **HV OUT CTRL0**，将它的 **OE0/1** 端子连接到 POR，并将其 Slew rate 设置为“**Fast for pre-drive**”, HV OUT mode 为“**Half-Bridge**”。
3. 将 **PIN7** 和 **PIN8** 配置为“**HIGH** and **LOW** side on”
4. 启用 **ACMP0H**。将其 **IN-** source 设置为“1920 mV”，从而把输出电压调节到 5 V。将其输出连接到 **2-bit LUT2** 的 **IN0** 端子。
5. 将 **CNT1/DLY1** 配置为 **Delay**，并将其输出连接到 **DFF10** 的 **nRESET** 端子和 **2-bit LUT2** 的 **IN1** 端子。将 **CNT1/DLY1** 的 DLY IN 端子连接到 **OSC1** 的 **Flex-Div OUT** 端子，配置 **OSC1** 属性中 Flexible divider 的值为 125。
6. 将 **DFF** 的 D 端子连接到 POR，将其 **CLK** 端子连接到 **2-bit LUT2** 的输出。
7. 将 **DFF** 的输出连接到 **HV OUT CTRL0** 的 **IN0** 和 **IN1** 端子。

应用：降压型转换器

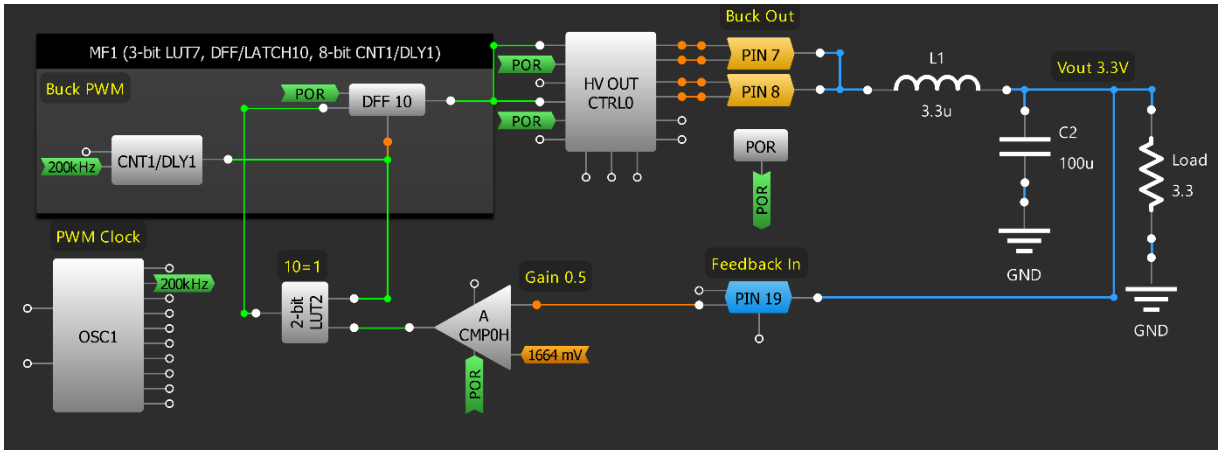
在本应用中，SLG47105 用作降压转换器。在本设计示例中，VDD2 电压被降至 3.3 V。此外，本设计还展示了如何构建模拟 PWM 模块。



所需元器件

- SLG47105
- 电感
- 3 个电容

GreenPAK 设计图

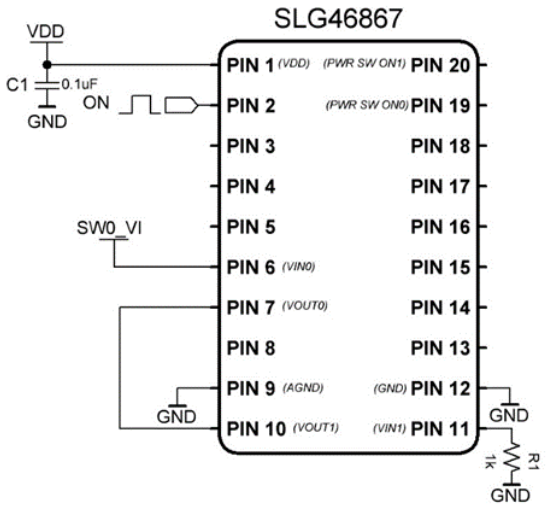


设计步骤

1. 电路外围配置:**PIN7** 和 **PIN8 同时连接 L1 的一端，PIN19 连接 L1 的另一端，PIN19 同时并入 C2 和负载电阻 load**。
2. 连接 **OE0/1** 到 **POR** 使能 **HV OUT CTRL0**，并将其 Slew rate 设置为“**Fast for pre-drive**”和 HV OUT mode 设置为“**Half-Bridge**”。
3. 配置 **PIN7** 和 **8** 为“**HIGH** and **LOW** side on”。
4. 通过连接 PWR_UP 到 **POR** 使能 **ACMP0H**。将其 **IN +** gain 设置为 x0.5，**IN -** source 设置为“1664 mV”，以调节输出电压至 3.3 V。将其输出连接到 **2-bit LUT2** 的 **IN0**。
5. 配置 **CNT1/DLY1** 为 Delay，并将其输出连接到 **DFF10** 的 **nRESET** 和 **2-bit LUT2** 的 **IN1**。将 **DLY IN** 连接到 **OSC1** 的 **Flex-Div OUT**。将 **Flexible** divider 设置为 125。
6. 将 **DFF** 的 **D** 连接到 **POR**，其 **CLK** 连接到 **2-bit LUT2** 的输出。
7. 将 **DFF** 的输出连接到 **HV OUT CTRL0** 的 **IN0** 和 **IN1**。

应用: 背对背的反向电流阻断

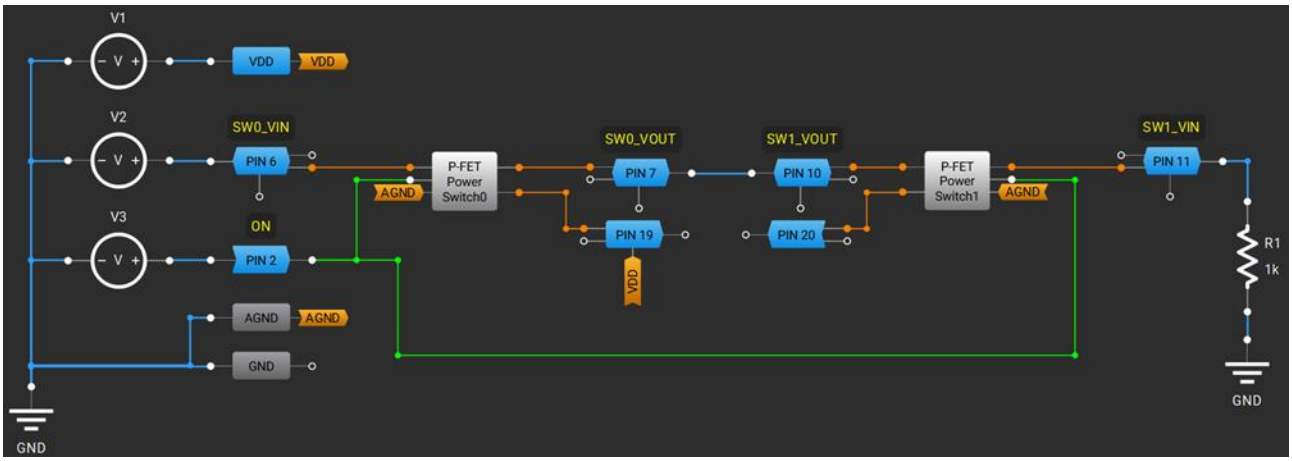
背对背的反向电流阻断是用于防止反向漏电流倒灌，一般会使用两个反向连接的 **MOSFET** 去阻断反向的电流。这种架构普遍使用在电源、电池驱动的终端和 **DC-DC** 转换器上，以提高效率及防止倒灌电流产生的损害。



所需元器件

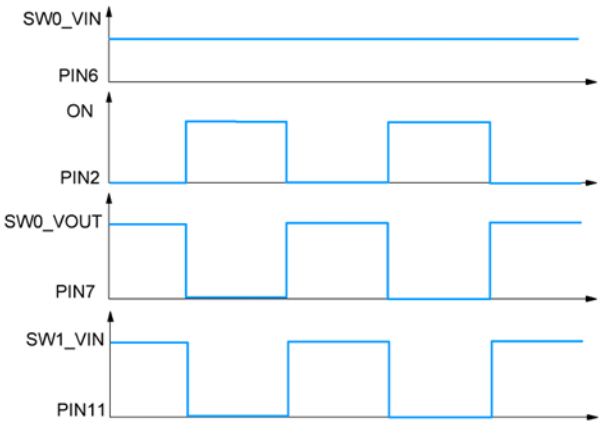
集成两个 **PMOS** 的 **GreenPAK**

GreenPAK 设计图



设计步骤

1. 配置 **P-FET Power Switch0** 和 **Switch1** 的控制模式为 “**Internal (ON input)**”
2. 连接 **PIN 7** 到 **PIN 10**
3. 连接负载 到 **PIN 11**

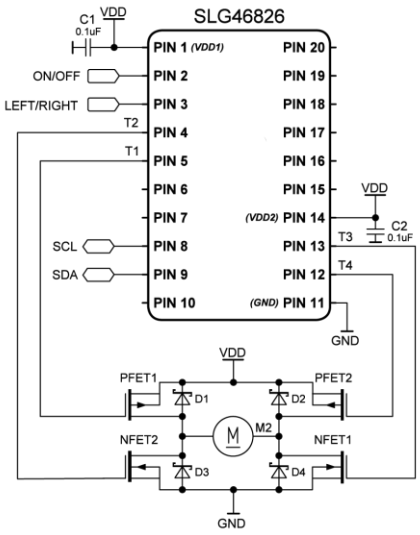


本章介绍直流电机控制的相关应用。着重于使用 SLG47105 的集成 H-Bridge 及其配套模块来提供电流和电压调节。

第 8 章:电机控制

应用：H 桥控制

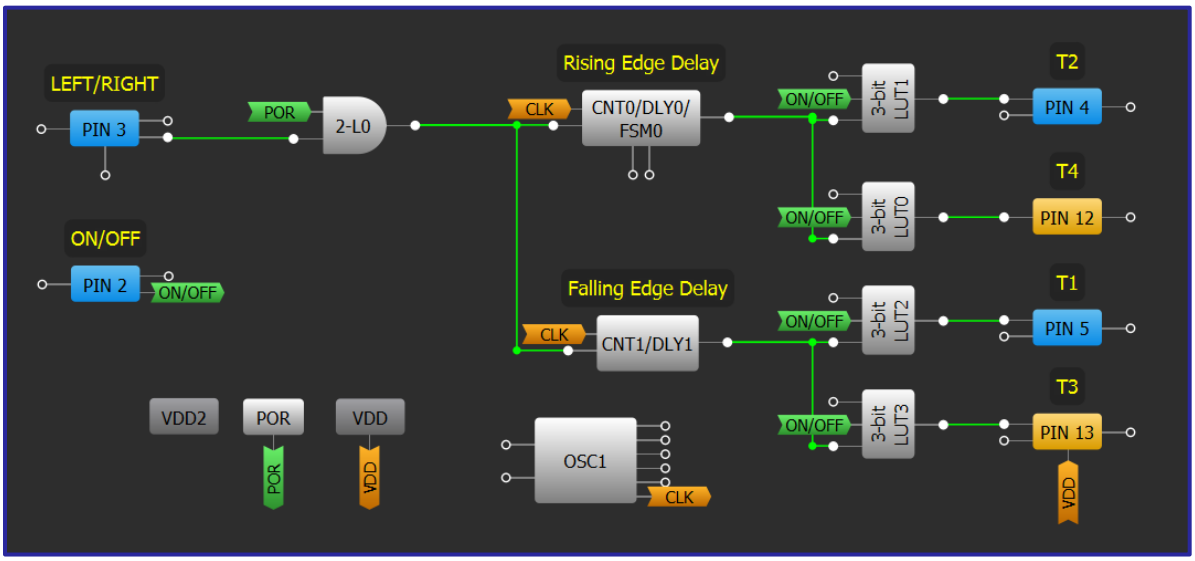
H 桥是一种由四个晶体管构成的电子电路，能够向负载施加反向电压。通常用于直流电机控制。



所需元器件

- 任意 GreenPAK IC(CMIC)
- 4 个晶体管
- 4 个二极管

GreenPAK 设计图



设计步骤

1. 添加并配置所需的输入引脚和输出引脚。
2. 使用 技巧：优化 CNT/DLY 的精度 添加和配置 2 个 DLY 组件，以设置精确的死区时间。
3. 使用 技巧：配置标准逻辑 为各个输出添加并配置 LUT。

技巧：使用 HV OUT CRTL 模块

*此技巧描述了 SLG47105V 中 **HV OUT CTRL** 模块的使用。*

SLG47105V 包含 2 个名为 **HV OUT CTRL0** 和 **HV OUT CTRL1 的** HV OUT CTRL 宏单元。每个宏单元可用于驱动 2 个单向直流电机或 1 个双向直流电机。**HV OUT CTRL0** 和 **HV OUT CTRL1** 均可用于驱动步进电机。

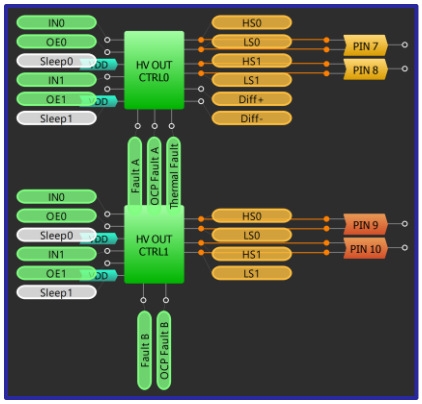


Figure 1. HV OUT CTRL0/1

要启用 **HV OUT CTRL0/1**，需将 **Sleep 0/1** 连接到有效的低电平。每个 Sleep 引脚都可以单独激活。驱动单向直流电机，选择 **HV OUT mode** 为“Half-Bridge”，驱动双向电机选择“Full-Bridge”。在“Full-Bridge”模式下，可将 **Mode control** 选择为“IN-IN”或“PH-EN”。表 2 显示了“Half-Bridge”控制逻辑。表 1 和表 3 分别描述了“IN-IN”和“PH-EN”模式控制。引脚 7/8/9/10（默认为 Hi-Z）用于使用 PWM 控制电机速度。这些引脚可以从外部上拉/下拉。在“Full Bridge”模式下，引脚 7/9 和引脚 8/10 可以在外部并联。

要使用 **PWM0/1** 通过“IN-IN”模式控制来控制直流电机，需将 **IN1** 连接到 LOW 信号时快速模式有效和连接到 HIGH 信号慢衰减模式有效。将 **IN0** 连接到 **PWM0/1** 输出。快速衰减模式用于立即降低感应电流并将电机滑行至零速度。慢衰减模式实现感应电流缓慢减少和快速减速。

Table 1. IN-IN Mode Logic for Full Bridge Mode

Sleep 0/1	IN0	IN1	Pin 7/9	Pin 8/10	Function
1	X	X	Hi-Z	Hi-Z	Off
0	0	0	Hi-Z	Hi-Z	Coast
0	0	1	L	H	Reverse
0	1	0	H	L	Forward
0	1	1	L	L	Brake

Table 2. Half-Bridge Mode Logic

Sleep 0/1	OE	IN0/1	Pin 7/8	Function
1	X	X	Hi-Z	Off
0	0	X	Hi-Z	Off (Coast)
0	1	0	L	Brake
0	1	1	H	Forward

Table 3. PH-EN Mode Logic for Full Bridge Mode

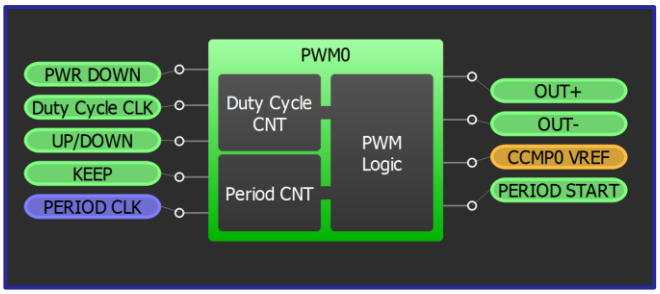
Sleep 0/1	Decay	EN/PWM	PH/Direct	Pin 7/9	Pin 8/10	Function
1	X	X	X	Hi-Z	Hi-Z	Off (Coast)
0	0 (fast)	0	X	Hi-Z	Hi-Z	Coast
0	1 (slow)	0	X	L	L	Brake
0	X	1	0	H	L	Forward
0	X	1	1	L	H	Reverse

发生任何故障时，**Fault A/B** 引脚变为高电平且 **HV OUT CTRL0/1** 被禁用。当故障引脚变为低电平时，恢复正常操作。**Fault A** 和 **Fault B** 分别包含 VDD2_A 和 VDD2_B 的所有故障信号。当发生过流情况时，**OCP Fault A/B** 变为高电平。可以在引脚 **7/8/9/10** 上启用过流保护的设置过流保护去抖时间(OCP deglitch time)。 **OCP deglitch time enable** 在引脚 **7/8** 和引脚 **9/10** 之间共用。 OCP 的恢复时间是用户可选择的，并且每个引脚都是独立的。当芯片温度超过安全限值时，**Thermal Fault** 变为高电平。 VDD2_A 和 VDD2_B 具有单独的 UVLO（欠压锁定）使能。

技巧：在常规模式下使用 SLG47105 PWM 模块

此技巧适用于SLG47105 中的PWM 模块。

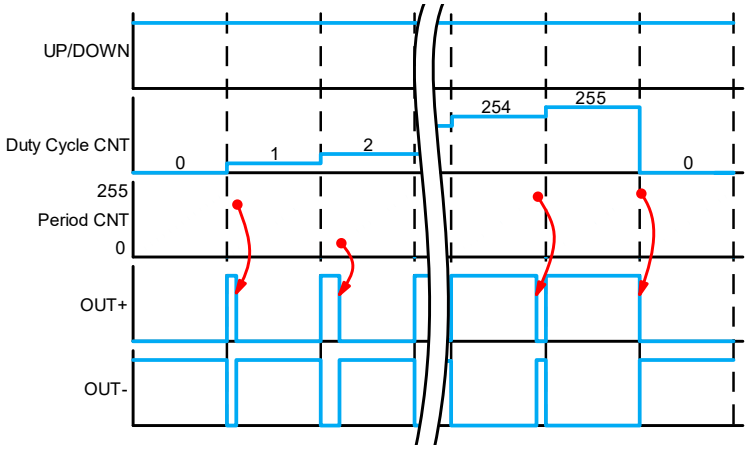
PWM 通常用于直流电机控制，LED 亮度控制及其他应用中。SLG47105 配备了先进的 PWM 模块来处理更高电压等级的专用 PIN。PWM 模块已在其他 GreenPAK 芯片中使用，如技巧：在 PWM 模式下使用 DCMP/PWM 宏单元，但 SLG47105 中的两个 PWM 模块在“常规模式”下集成了与其操作相关的计数器，并且包含更高级的设置。



该块中包含的第一个 8 位计数器是 **PWM Period CNT**，它可设置 PWM 信号的频率，从 0 计数到 255 等等。该有两个 PWM 输出：“OUT+”和“OUT-”。 OUT+ 在周期开始时为逻辑高电平，一旦 **PWM period counter** 达到占空比值，输出变为逻辑低电平，直到 PWM 周期结束，如下图所示。 **OUT+** 是正 PWM 输出，**OUT-** 是负 PWM 输出，它与 **OUT+** 反转并在死区时间（如果定义）内移位。两者都可以通过寄存器设置反转它们的输出。

在常规模式下（如此节所述），**Duty cycle source** 设置为 Duty Cycle CNT。包含的第二个 8 位计数器，称为 **Duty Cycle CNT**，根据 **UP/DOWN** 输入递增或递减下一个 PWM 周期的占空比值。Duty Cycle CLK 默认是来自矩阵的外部时钟。它通过上升沿改变占空比值。它也可以设置为由周期计数器溢出或每第 2 或第 8 个脉冲的溢出提供时钟。

默认情况下，PWM 模块具有 8 位分辨率，但也可以选择 7 位分辨率以允许更高的 PWM 频率。对于 8 位分辨率，PWM 占空比以 0.4% 的步进变化，而对于 7 位分辨率，则以 0.8% 的步进变化。占空比可以从真正的 0% 变为真正的 100%。 PWM 从初始占空比值开始。该模块有一个 **UP/DOWN** 内部连接，用于定义占空比变化的方向。如果设置为 HIGH，占空比将逐步增大，如果设置为 LOW，则占空比将逐步减小。



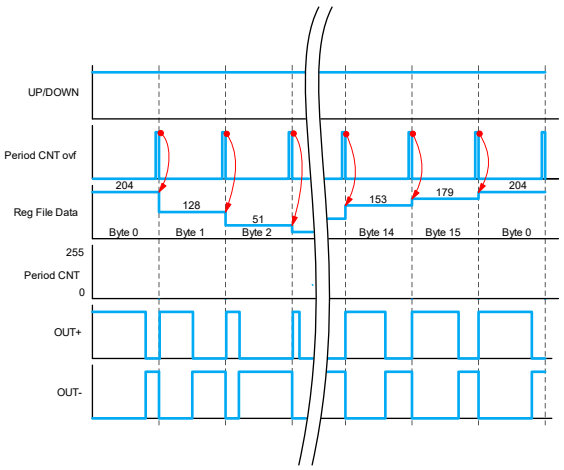
可以选择 **Keep/Stop** 连接以保持占空比（“Keep”设置）或在设置为高电平时保持占空比和 OUT+ 和 OUT- 输出恒定（“Stop”设置）。**Continuous/Autostop** 模式要么设置为“Continuous”，PWM 输出占空比在达到满量程值（默认设置）时溢出，要么设置为“Autostop”，PWM 输出在达到 0% 或 100% 占空比时停止。选择“Autostop”时，可以激活“Boundary OSC disable”选项。这允许在达到 0% 或 100%占空比时，自动禁用 PWM 单元使用的振荡器。

技巧：在预设寄存器模式下使用 SLG47105 PWM 模块

此技巧适用于SLG47105 中的PWM 模块。有关该模块如何工作的更多信息，请参阅技巧：在常规模式下使用 SLG47105 PWM 模块。

之前的技巧解释了如何在“常规模式”下使用 SLG47105 PWM 模块，并将 **Duty Cycle Source** 设置为“Duty Cycle CNT”。此技巧将改为解释如何在“预设寄存器模式”下使用该模块，其中占空比循环通过 16 个预定义值（Reg File）。使用“预设寄存器模式”允许产生一组非线性的 PWM 控制电机（即正弦或对数）。

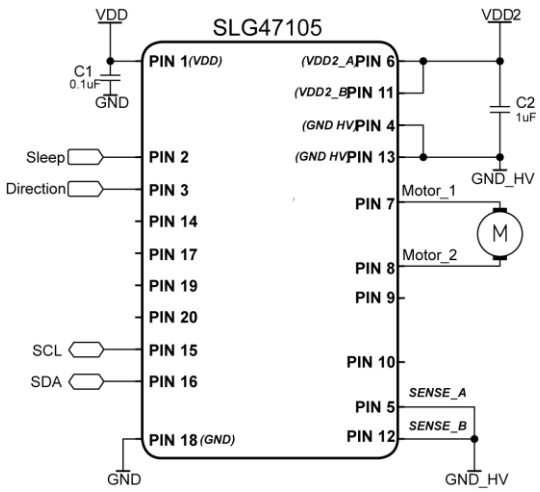
保留可选择的预设寄存器以定义 16 个不同的 PWM 占空比值。Duty Cycle CLK 可以选择 Matrix 或 PWM period CNT ovf （溢出）为时钟。**Duty Cycle CNT CLK** 输入的时钟会改变了其寄存器的值与 Period CNT 进行比较。 Reg 文件在两个 PWM 宏单元之间共用。可以使用全部 16 个字节、较低 8 个字节有效或较高 8 个字节有效。初始字节受每个设置的唯一范围限制。



内部连接具有与常规模式类似的功能，但适用于 16 字节结构而不是 8 位计数器。 Up/Down 的极性决定是应用下一个寄存器（HIGH）还是上一个寄存器（LOW）。 Keep/Stop 的操作方式相同，但会暂停寄存器序列。

应用：恒压有刷直流电机驱动器

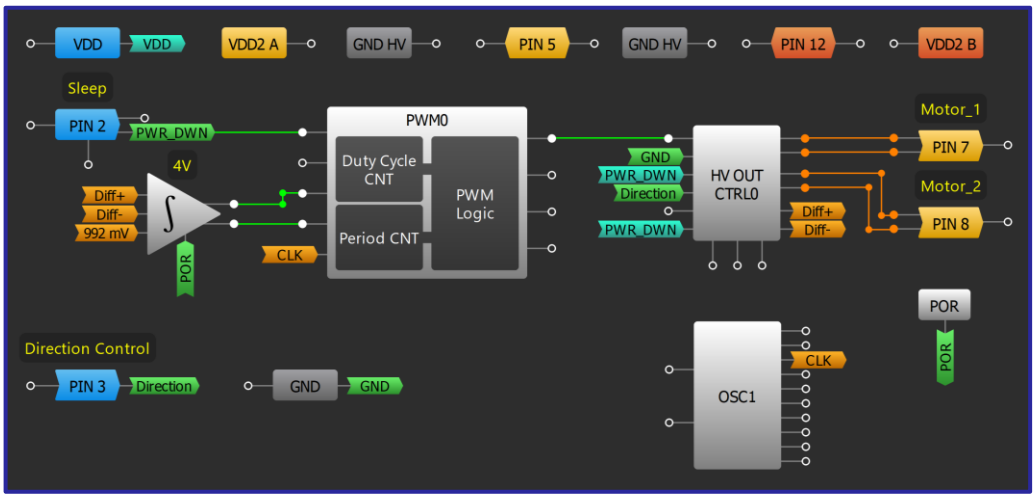
在有刷直流电动机上保持恒定的电压可以确保它保持恒定的速度。 在本设计中，集成积分器和比较器的差动放大器控制PWM 模块来调节负载电压。



所需元器件

- SLG47105V
- 有刷直流电机

GreenPAK 设计图

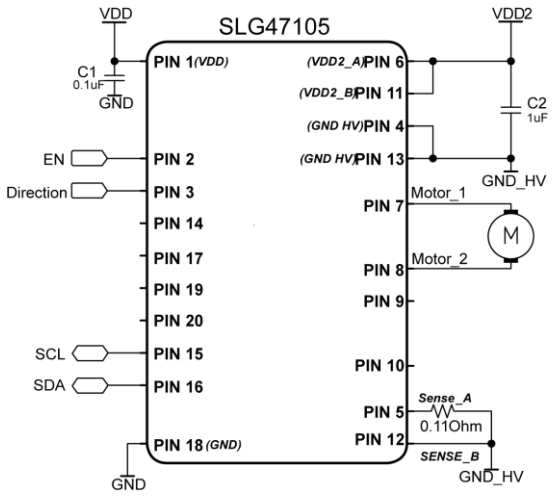


设计步骤

1. 通过 **PIN2** 启用 **HV OUT CTRL0**，并设置其 Mode 为“Full bridge”，Mode control 为“PH-EN”。
2. 要改变电机方向，请将 **PIN3** 连接到 **HV OUT CTRL0** 的 **PH** 脚。
3. 使能集成了积分器和比较器的差动放大器，并选择积分器参考电压为所需的阈值(threshold =V_REF·4)。
4. 为了使集成积分器和比较器的差动放大器正确运行，通过 **PIN2** 使能 **PWM0**，并将 **Duty** Period CLK 设置为 “**OSC1**”。 PWM 频率必须为 44khz 或更高，以确保积分器正常工作。将 UPWARD 和 Equal 输出分别连接到 PWM0 的 UP/DOWN 和 Keep。
5. 设置 PWM0 Resolution 为“8-bits”，Duty Cycle Source 为“Duty Cycle CNT”，Duty Cycle CLK 为“Period CNT ovf/8”，Initial Duty Cycle Value 为“50%”。
6. 将 **PWM0 OUT+**与 **HV OUT CTRL0 的 EN** 脚连接，以恒速驱动电机。

应用：恒流有刷直流电机驱动器

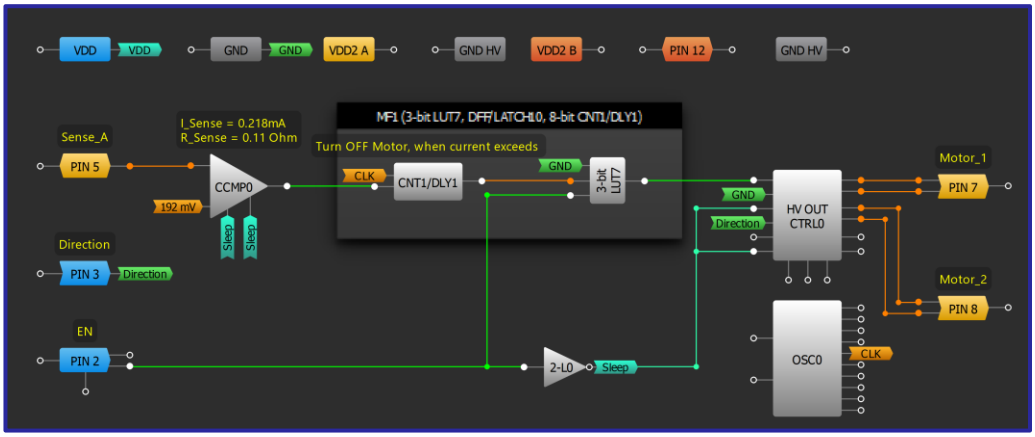
在有刷直流电动机上保持恒定的电流可以确保它保持恒定的转矩。 在本设计中，电流检测比较器(CCMP0)用于限制通过采样电阻的电流，当电流超过规定的限制值时，关闭电机。



所需元器件

- SLG47105V
- 有刷直流电机
- 1 个电阻器

GreenPAK 设计图

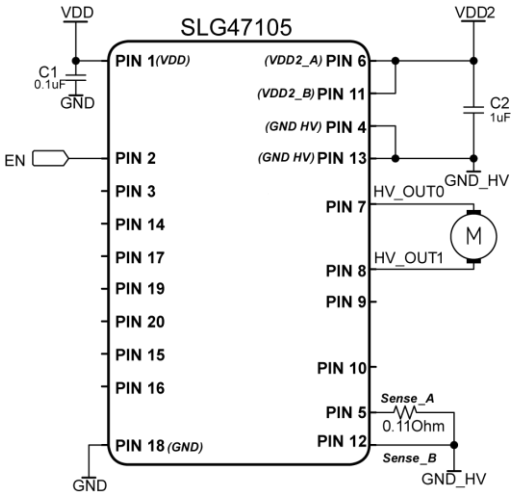


设计步骤

1. 通过 **PIN7** 和 **PIN8** 连接有刷直流电机，在 **PIN5** 与 GND 连接采样电阻。
2. 将 **PIN2 的反转**信号连接到其 **Sleep 0/1**，使能 **HV OUT CTRL0**。
3. 配置 **HV OUT CTRL0** 的模式为“全桥”，**模式控制**为“PH-EN”。
4. 为了改变电机方向，连接 **Pin 3** 到 HV OUT CTRL0 的 **PH** 脚。
5. 将 **Sleep CTRL** 更改为“Auto”来启用 **CCMP0**。
6. 设置 CCMP0 的 **IN- source** 为“192mV”，以限制电流为 0.218mA。
7. 配置 **CNT1/DLY1**，当电流超过规定的限制值时，关闭电机 100ms。
8. 配置 **3-bit LUT7**，只有在电流在范围内且 **PIN2** 为 HIGH 时才启动电机。
9. 将 **3-bit LUT7** 输出连接到 **HV OUT CTRL0** 的 **EN** 脚上。

应用：PWM 恒流斩波器

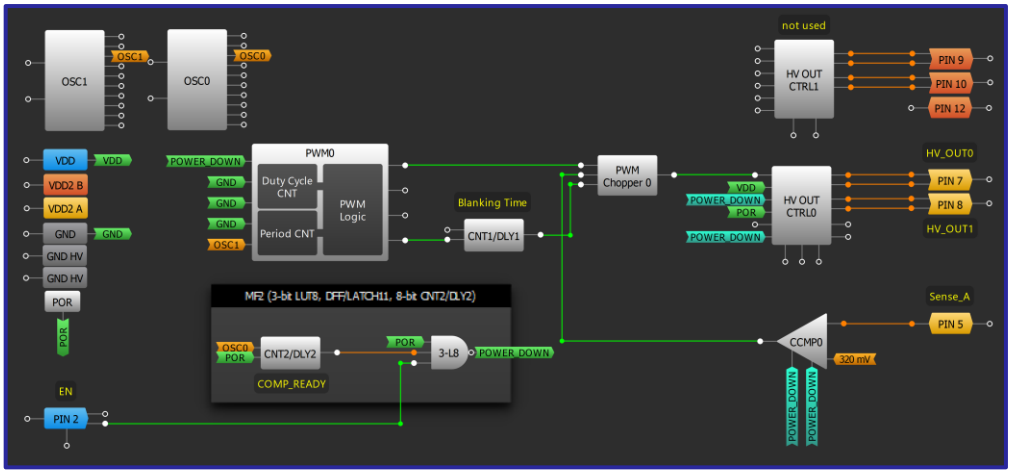
在有刷直流电动机上保持恒定的电流可以确保它保持恒定的转矩。 这个应用程序展示了如何用 PWM 斩波器限制电流。



所需元器件

- SLG47105V
- 有刷直流电机
- 1 个电阻器

GreenPAK 设计图

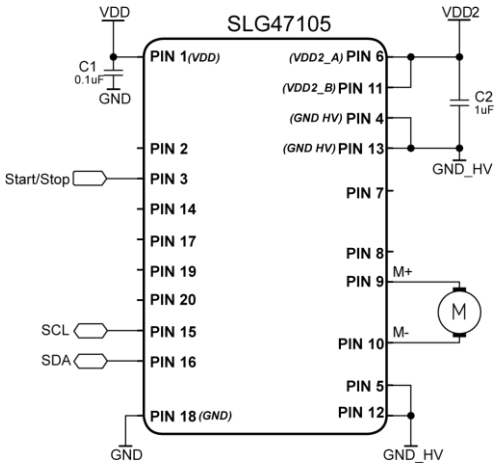


设计步骤

1. 通过 **PIN7** 和 **PIN8** 连接有刷直流电机，在 Pin5 与 GND 间连接采样电阻。
2. 配置 HV OUT CTRL0 模式为“Full bridge”，**模式控制**为“PH-EN”。
3. 更改 **Sleep CTRL** 为“Auto”以启用 **CCMP0**。
4. 选择 CCMP0 的 **IN- source** 为“320mV”以限制电流~0.36mA。
5. 添加 **PWM0** 模块，初始占空比为 230，调整 **OSC1** 预分压器。
6. 将 **CNT1/DLY1** 设置为下降沿延时，以设置“消隐时间”。
7. 添加 **PWM 斩波器 0**，并与 **PWM0、CNT1/DLY1 和 CCMP0** 进行适当的连接，以创建占空比斩波器并限制电机电流。
8. 添加 **Pin2** 作为启动按钮，启动/停止电机和内部电机控制模块。
9. 将 **3-bit LUT7** 输出连接到 **HV OUT CTRL0** 的 EN 脚上。

应用：带有软开关的单向直流电机控制

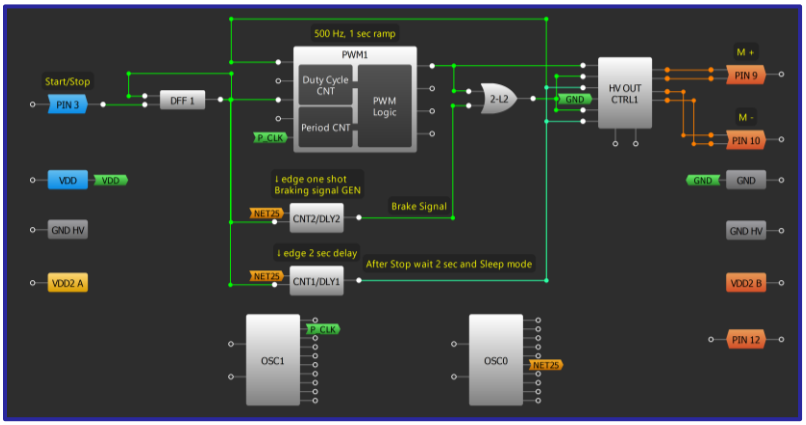
软开/关可以降低有刷直流电机启动时的电流和负载转矩。 这个应用程序是单向直流电机，允许一个单一的方向的机械元件，或运行或停止。 这种调节是重要的，因为负载转矩启动和停止不能超过电机轴上的转矩。

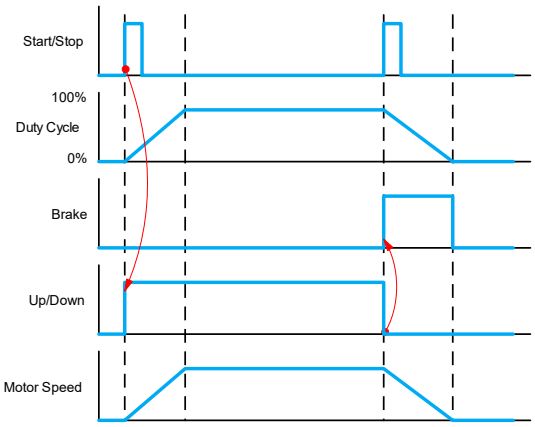


所需元器件

- SLG47105
- 单向有刷电机
- 2 个电容

GreenPAK 设计图



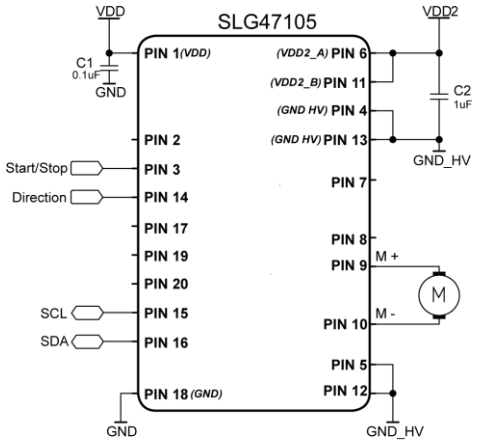


设计步骤

1. 设置“**PIN9**”和“**PIN10 Output mode**”为“High and Low side”。 **PIN10 Output mode** 设置为“Half Bridge” **HV OUT mode** 。
2. 将 **PWM1** 块的 **Duty Cycle CLK** 设置为“Period CNT ovf/2”。 **Period CLK** 设置为“Ext.Clk”。 连接 OSC1 Flex-DIV OUT 到 PWM1 周期时钟输入。 在 OSC1 属性中设置 flexible divider 的值。
3. 配置 **DFF1 Initial Polarity** 为“HIGH”， **Q Output Polarity** 为“Inverted (nQ)”。 将 DFF 的输出连接到 **PWM1 UP/DOWN 输入**。
4. 加入 **CNT1/DLY1** 进行设计。 将其配置为反向输出的“Falling Edge Delay”。 来自这个延迟的信号使 **PWM1** 块失效，并在停止信号到来后使能 **HV OUT CTRL1** 的睡眠模式。
5. **2-bit LUT2** 用于形成 HV Outputs 脚的 Hi-Z 和停止信号。

应用：具有软开/关的双向直流电机控制

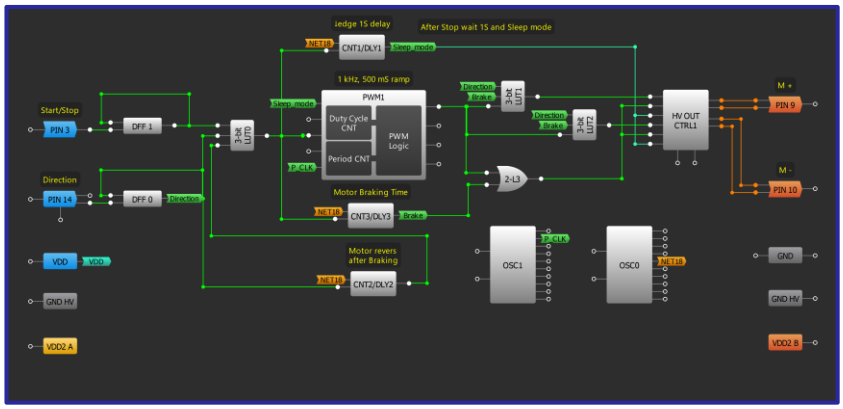
软开关可用于降低有刷直流电机启动时的电流和负载转矩。此应用适用于机械元件可以反向运动的双向电机。电机制动扭矩必须超过负载扭矩才能使转动停止，然后才能反向转动。

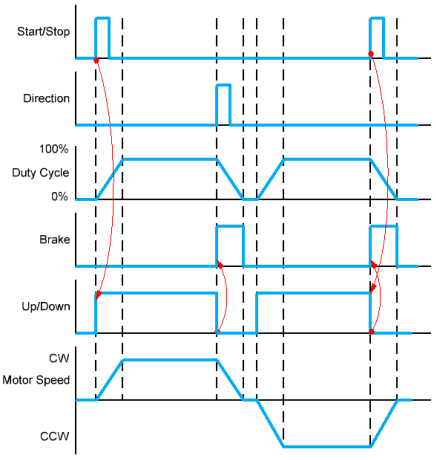


所需元器件

- SLG47105
- 双向有刷电机
- 一颗电容器

GreenPAK 设计图

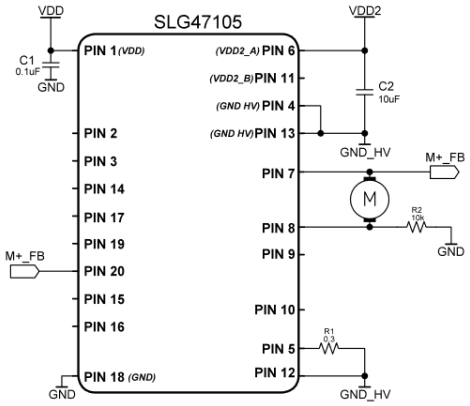




设计步骤

1. 将 **PIN9** 和 **PIN10** 的 **Output mode** 配置为 “High and Low side.” **HV OUT CTRL1** 配置为 “Half Bridge” **HV OUT mode**。
2. 将 PWM1 模块的 **Duty Cycle CLK** 配置为 “Period CNT ovf/2.” **Period CLK** 配置为 “Ext.Clk.” 将 OSC1 Flex-DIV OUT 连接到 PWM1 的 Period Clock 输入。配置 OSC1 属性中 flexible divider 的值。
3. 将 **DFF1 的 Initial Polarity** 配置为 High，**Q Output Polarity** 为 Inverted (nQ)。**DFF0** 也进行相同的配置。将 DFF 的输出连接到 **3-bit LUT0**。
4. 将 **CNT1/DLY1** 配置为 Falling Edge Delay 和 inverted output。来自该延时的信号将禁用 PWM1 模块并在 Stop 信号到来后启用 **HV OUT CTRL1** 的睡眠模式。
5. 将 **CNT2/DLY2** 配置为 Both Edge delay，以设置在反转前电机停止的延时。
6. 将 **CNT3/DLY3** 配置为 Falling edge One Shot, 将 counter 的输出连接到 **3-bit LUT1** 和 **3-bit LUT2**。

应用：助推启动/堵转停止

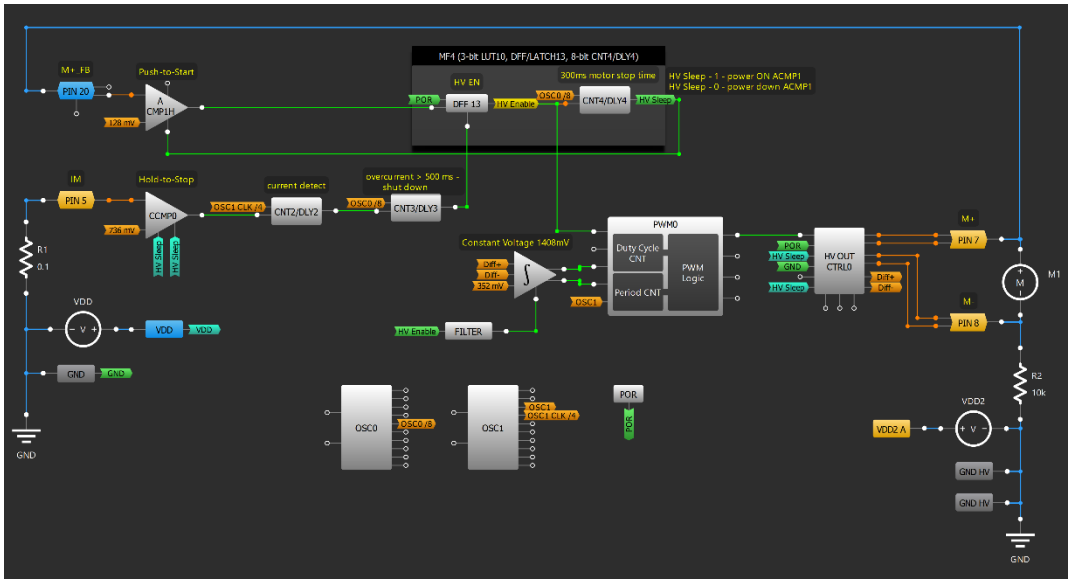


本应用展示了 SLG47105 在有刷直流电机应用中实现助推启动、按压停止的功能。该特性可用于电子玩具、电动工具和其他应用场景。

所需元器件

- SLG47105
- 有刷直流电机
- 2 个电阻器

GreenPAK 设计图

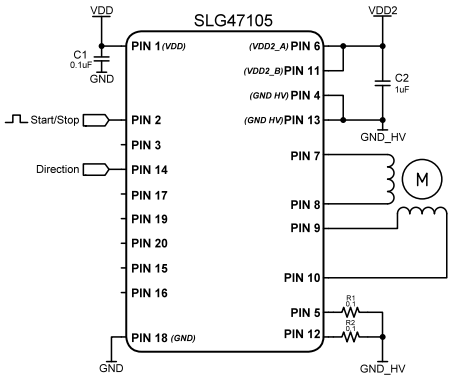


设计步骤

1. 按如下步骤连接**有刷直流电机**：绕组 1 连接到 **PIN7**, 绕组 2 连接到 **PIN8** 并连接 10k 的下拉电阻，在 **PIN5** 与 **GND** 间连接采样电阻。**PIN 7** 和 **PIN 8** 的 Output mode 设置为 “HIGH and LOW side”。
2. 启用 **HV OUT CTRL0**，将它的 Sleep 0/1 端子连接到有效的低电平(**CNT4/DLY4** 输出脚 HV Sleep)，并设置其 HV OUT mode 为 “Full Bridge”，Mode controls 为 “PN-EN”。将 Decay 端子连接到 POR、PH 端子连接到 GND。
3. 更改 Sleep CTRL 为“Auto”以启用 **CCMP0**，设置 **CCMP0** 的 IN+ gain 为“x4”, IN- source 为 “736 mV” 以实现“堵转停止”功能。
4. 将 **CNT2/DLY2** 配置为“Delay”，并将其输出脚连接到 **CNT3/DLY3** (Mode 为 Delayed edge detect)，以保证系统在检测到过流 500 ms 后关闭 – 实现堵转停止。
5. 将 **PIN20** 配置为“Analog input/output”并连接到 **ACMP1H** 的 IN+ source，选择 IN– source 为“128mV”，并连接比较器的输出脚至 **DFF13** 的 CLK(其 Q Output Polarity 配置为 Inverted (nQ) , Initial Polarity 配置为 Low) – 实现助推启动。
6. 将 **CNT4/DLY4** 配置为(Rising)Delay，以保证 300 ms 的电机停转时间(HV Sleep 信号)。
7. 添加 **PWM0** 模块并与**差分放大器** (Integrator reference volt 设置为 352 mV) 进行适当的连接，以确保有刷直流电机的驱动电压保持恒定值 **(参考应用示例：恒压有刷直流电机驱动器)**, 与 **DFF13 和 HV OUT CTRL0** 连接以获得需要的占空比。

应用：双极步进电机驱动器

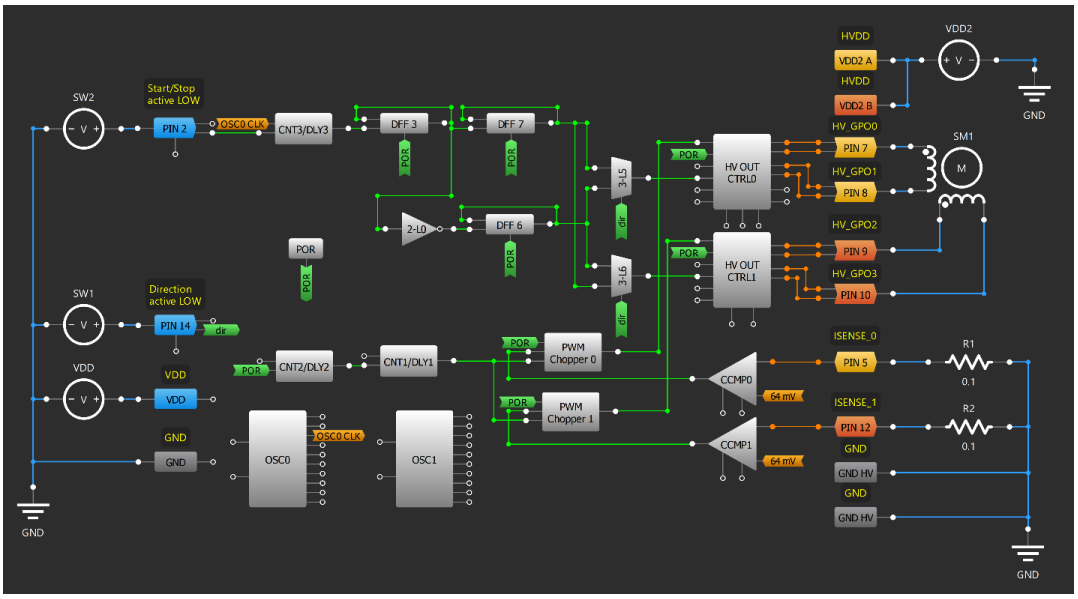
本应用展示了利用 SLG47105 作为步进电机驱动器的方法。在本设计实例中，驱动器支持全步双极性驱动。同时，本设计也展示了如何通过 PWM 斩波器实现限流功能



所需元器件

- SLG47105
- 步进电机
- 2 个电阻器

GreenPAK 设计图



设计步骤

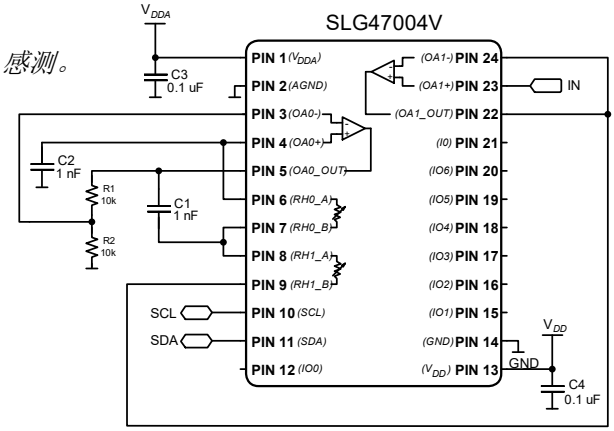
1. 按以下步骤连接步进电机:
   - 通过 **PIN7** 和 **PIN8** 连接绕组 1, 在 **PIN5** 与 GND 连接采样电阻；
   - 通过 **PIN9** 和 **PIN10** 连接绕组 2, 在 **PIN12** 与 GND 连接采样电阻。
2. 将 **HV OUT CTRL0/1** 的 Sleep 0/1 端子连接到有效的低电平，并设置其 HV OUT mode 为 “Full Bridge”，Mode controls 为 “PN-EN”。
3. 更改 Sleep CTRL 为“Auto”以启用 **CCMP0/1**，设置 **CCMP0/1** 的 IN- source 为 “64 mV” 以限制电流为 80 mA.
4. 将 **CNT2/DLY2** 配置为 Reset Counter, 并将其输出端子 OUT 脚连接到 **CNT1/DLY1** (Mode 为 Delay) 的输入端子 DLY_IN 以设置“消隐时间”。
5. 添加 **PWM0/1 Chopper** 模块并与 **POR**、**CNT1/DLY1** 和 **CCMP0/1** 进行适当的连接，以创建占空比斩波器并限制电机电流，将其输出连接至 **HV OUT CTRL0/1** 的 EN 端子。
6. 将 **PIN2** (Start/Stop) 和 **PIN14** (Direction) 配置为“Digital Inputs”并与以下组件进行适当的连接：**CNT3/DLY3** 的 Delay 脚、**DFF3**、**DFF7**、**DFF6** (Q Output Polarity 设置为 Inverted (nQ), Initial Polarity 设置为 Low)、**2-bit LUT0**（配置为 Invertor）、3 位多路复用器 **LUT5**、**LUT6**。为了控制电机启动/停止和改变电机转向，请将 LUT5/6 的输出脚连接到 **HV OUT CTRL0/1** 的 PH 端子。

第九章：高级模拟功能

本章展示了利用 AnalogPAK 中的组件实现模拟功能的应用。应用涉及使用内置运算放大器、数字变阻器、斩波比较器等最常见的电路拓扑。

应用：采用 OpAmp 可调有源滤波器

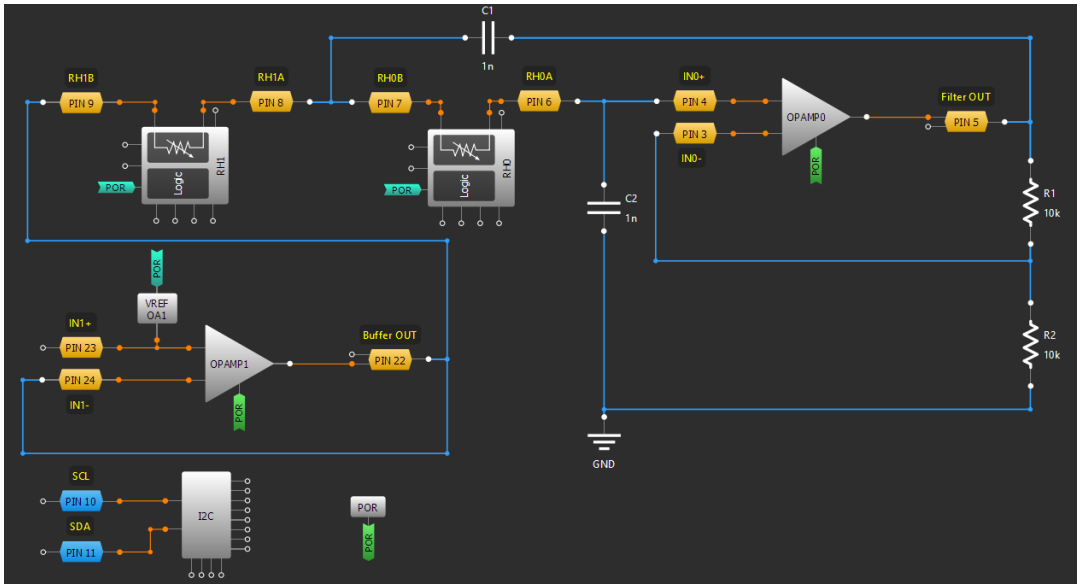
在许多应用中，来自不同来源(即传感器)的信号可以用一个 ADC 感测。这些系统需要一个模拟多路复用器，每个通道都有模拟滤波器，因为每个信号源可能有自己的一组滤波器要求(例如截止频率)。在本设计中，一个滤波器服务于多个模拟输入，并为它们提供不同的截止频率。I²C 主控可以将数据写入变阻器寄存器并调节滤波器的截止频率。



所需元器件

- SLG47004V
- 2 个电阻
- 2 个电容

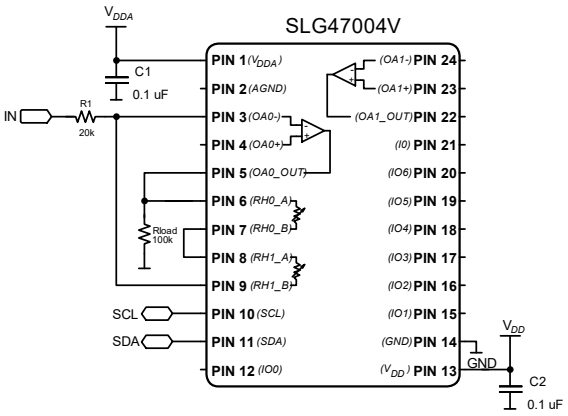
GreenPAK 设计图



设计步骤

1. 启用 **OpAmp0** 和 **Opamp**，设置带宽(bandwidth selection)为 8 MHz，并启用电荷泵(Charge Pump 为 Enable CP)。设置 **OpAmp1** Vref 连接到 IN+(Vref Connection 为 to IN+)。
2. 启用 **Vref OA1**，设置输入电压(Input Voltage selection)为 V_{DD}，输出(Output Selection)选择为 V_{DD} * (16/64)。
3. 设置 **Digital Rheostat 1** 为变阻器模式(Mode 为 Rheostat)，Auto-Trim 停用(Disable)，将 FIFO nRST 端子输入连接到 POR 并为全部电阻器设置所需电阻（初始数据(配置 Resistance initial)）。

应用：可调反相 OpAmp

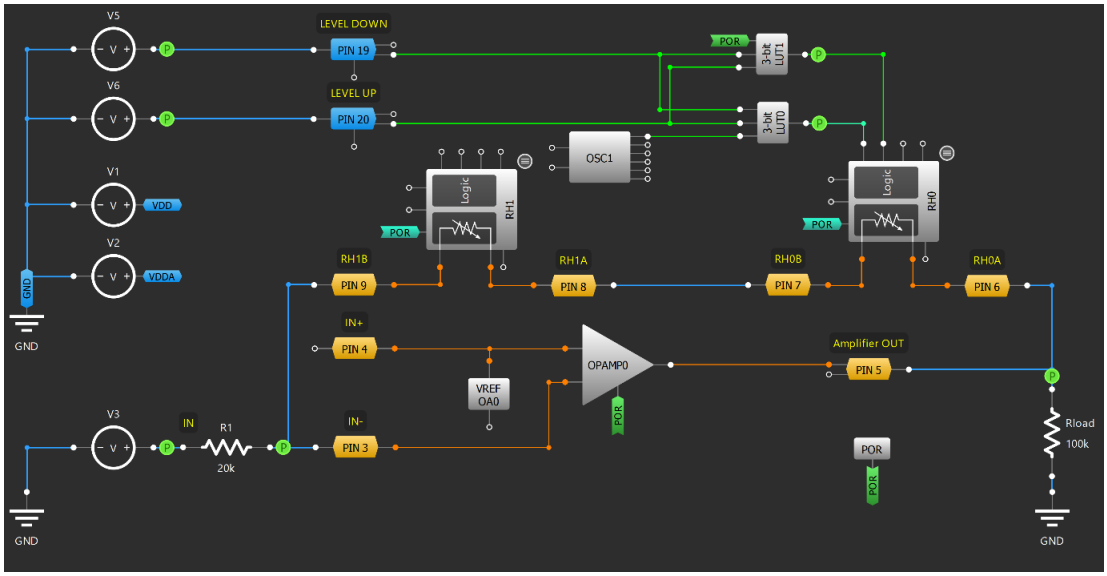


反相放大器是运算放大器电路的一种，它产生的输出与输入移相 180º。这意味着如果输入脉冲是正的，输出脉冲将是负的，反之亦然。外部控制信号允许用户双向调整反相放大器的增益。

所需元器件

- SLG47004V
- 1 个电阻

GreenPAK 设计图



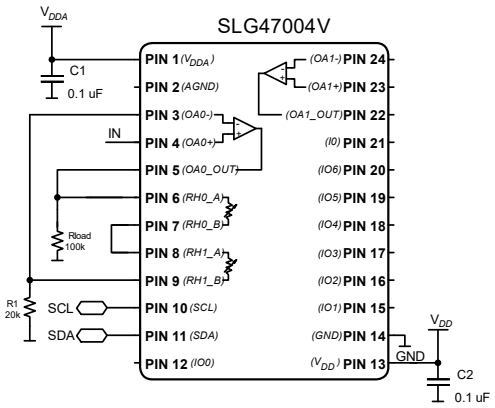
$$G = \frac{V_{out}}{V_{inp}} = -\frac{RH0+RH1}{R_1}.$$

设计步骤

1. 启用 **OpAmp0**，设置带宽(bandwidth selection)为 8 MHz，并启用电荷泵(Charge Pump 为 Enable CP)，设置 Vref Connection 为 to IN+。
2. 启用 **Vref OA0**，设置输入电压(Input Voltage selection)为 2.048 V，输出(Output Selection)选择为 512 mV。
3. 设置 **Digital Rheostat 1** 模式(Mode)为变阻器(Rheostat)，Auto-Trim 停用(Disable)，将 FIFO nRST 端子输入连接到 POR 并为全部电阻器设置所需电阻（初始数据(配置 Resistance initial)）。
4. 当输入 PIN19(LEVEL UP)和 PIN20(LEVEL DOWN)不同时为高或者同时为低，配置 **LUT0** 输出为 **OSC1** 时钟信号。当输入 PIN19(LEVEL UP)电平信号为高且 PIN20(LEVEL DOWN)为低时，配置 LUT1 输出高电平，否则 **LUT1** 输出低电平。

应用：可调同相 OpAmp

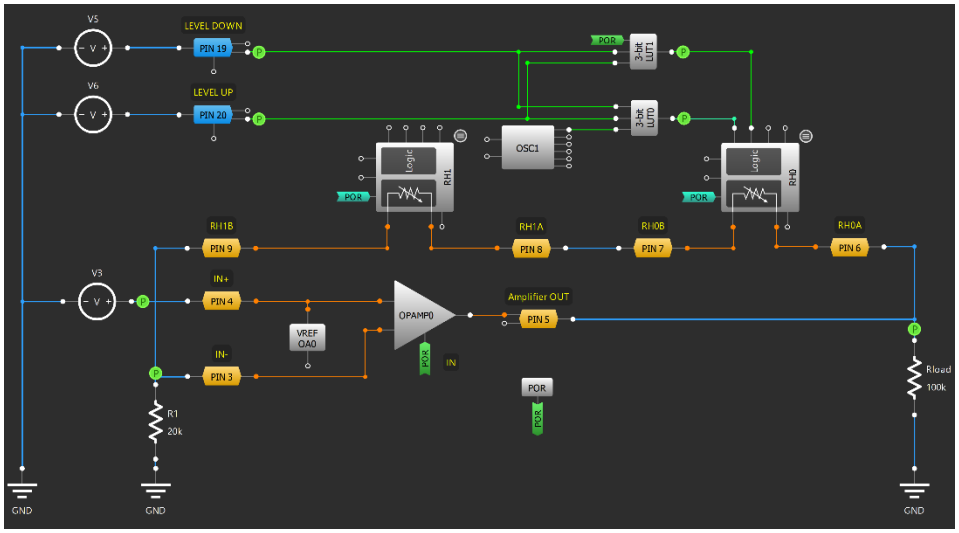
同相放大器是一种基于运算放大器的、具有正电压增益的放大器。当您将任何信号应用于同相输入端，当它在输出端被放大时，极性不会改变。所以，在这种情况下，放大器的增益总是正的。外部控制信号允许用户双向调整同相放大器的增益。



所需元器件

- SLG47004V
- 1 个电阻

GreenPAK 设计图



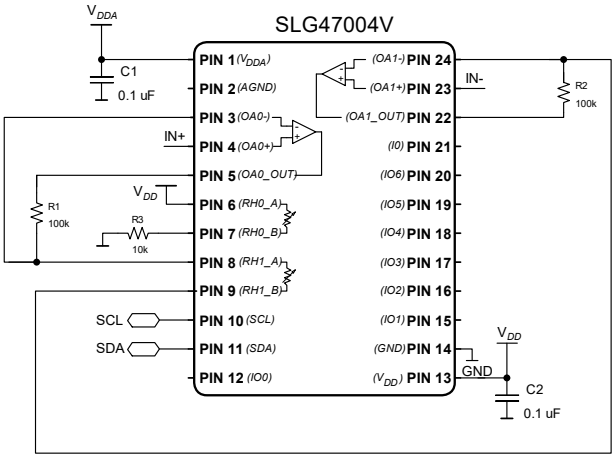
$$G = \frac{V_{out}}{V_{inp}} = (1 + \frac{RH0+RH1}{R_1}).$$

设计步骤

1. 启用 **OpAmp0**，设置带宽(bandwidth selection)为 8 MHz，并启用电荷泵(Charge Pump 为 Enable CP)，设置 Vref Connection 为 to IN+。
2. 启用 **Vref OA0**，设置输入电压(Input Voltage selection)为 2.048 V，输出(Output Selection)选择为 256 mV。
3. 设置 **Digital Rheostat 1** 模式(Mode)为变阻器(Rheostat)，Auto-Trim 停用(Disable)，将 FIFO nRST 端子输入连接到 POR 并为全部电阻器设置所需电阻（初始数据(配置 Resistance initial)）。
4. 当输入 PIN19(LEVEL UP)和 PIN20(LEVEL DOWN)不同时为高或者同时为低，配置 **LUT0** 输出为 **OSC1** 时钟信号。当输入 PIN19(LEVEL UP)电平信号为高且 PIN20(LEVEL DOWN)为低时，配置 LUT1 输出高电平，否则 **LUT1** 输出低电平。

应用：仪表放大器

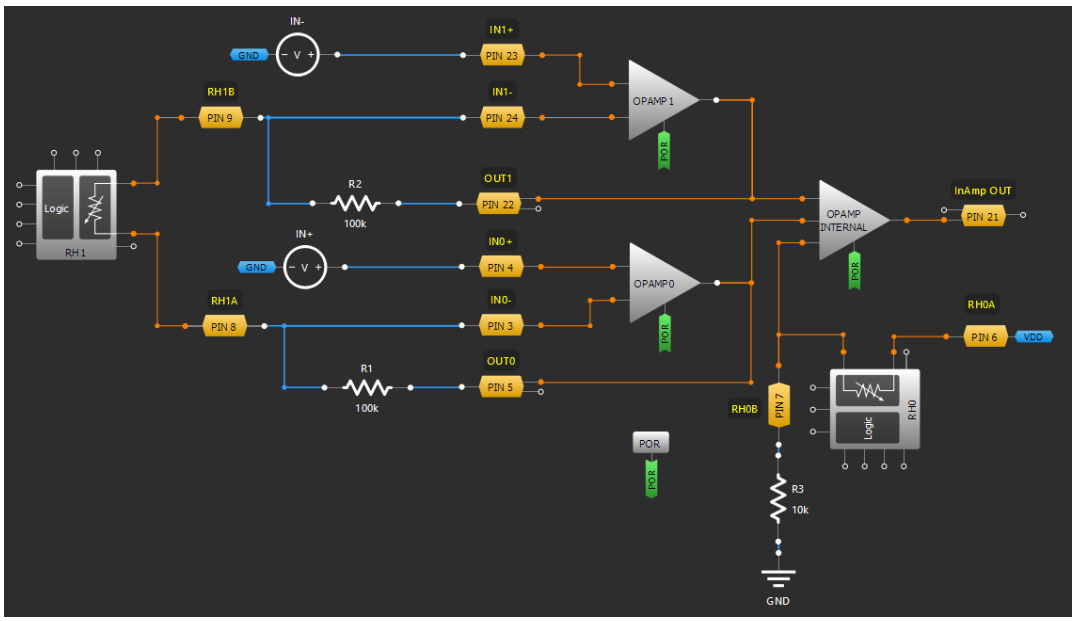
仪表放大器是一种配备了输入缓冲放大器的差分放大器，消除了输入阻抗匹配的需要。其他特性还包括极低的直流偏置、低漂移、低噪声、极高的开环增益、极高的共模抑制比和极高的输入阻抗。



所需元器件

- SLG47004V
- 3 个电阻

GreenPAK 设计图



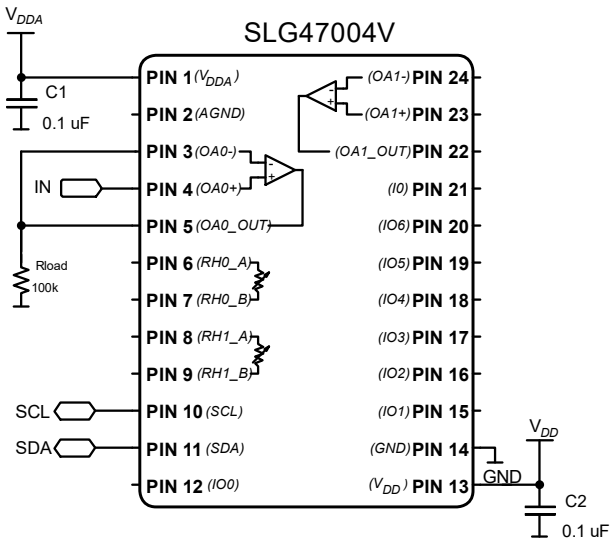
$$V_{OUT} = (1 + \frac{R_1+R_2}{R_{H1}})(V_{IN+} - V_{IN-}) + V_{DD}\frac{R_3}{R_{H0}+R_3}.$$

设计步骤

1. 使能 **OpAmp0**, **OpAmp1** 和 **OpAmp Internal**。设置带宽(bandwidth selection)为 128 kHz 并启用 Charge Pumps(Charge Pump 为 Enable CP)。**OpAmp Internal** 的 VREF Source 设置为 RH0 PIN B 。
2. 将 **Digital Rheostat 1** 设置为“Rheostat mode”。Auto-Trim 停用(Disable)，将 FIFO nRST 端子输入连接到 POR，并为全部变阻器设置所需电阻(初始数据(配置 Resistance initial))。

应用：使用运放创建电压跟随器

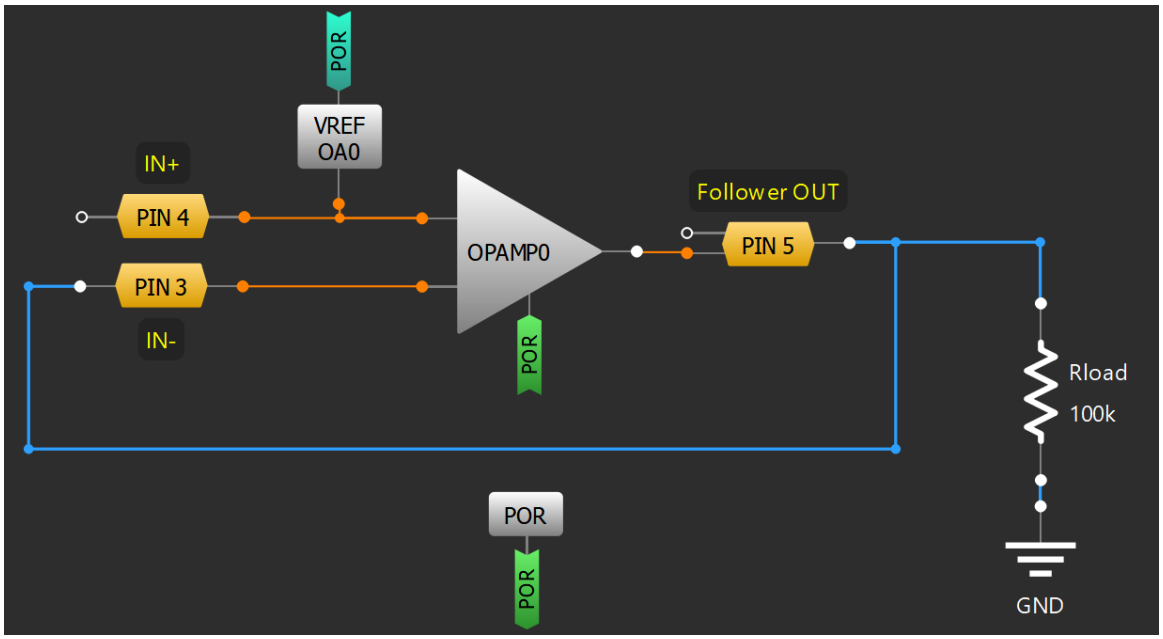
电压跟随器（也称为缓冲放大器）是一种输出电压等于输入电压的运放电路。因此，电压跟随器中的运放不会放大输入信号，其电压增益是 1。电压跟随器电路具有非常高的输入阻抗。这一特性使其常用于需要在输入和输出信号之间进行隔离的各种电路中。



所需元器件

- SLG47004V

GreenPAK 设计图

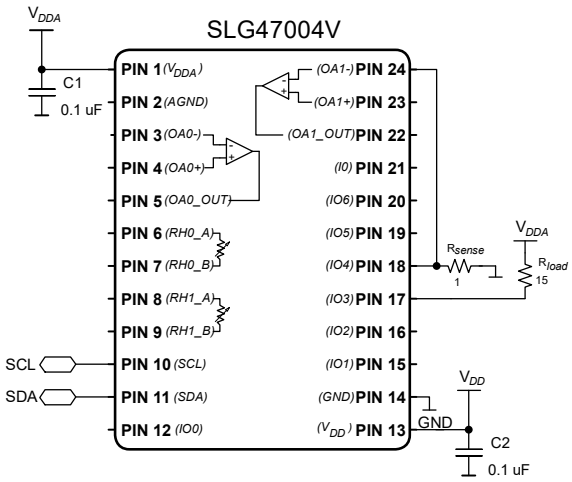


设计步骤

1. 启用 **OpAmp0**，设置带宽(bandwidth selection)为 8 MHz，并启用电荷泵(Charge Pump 为 Enable CP)，设置 Vref Connection 为 to IN+。
2. 启用 **Vref OA0**, 将 input voltage 设置为 VDDA，output selection 设置为 VDDA*(8/64)。

应用：使用运放和 N 沟道 FET 创建电流阱(Current Sink)

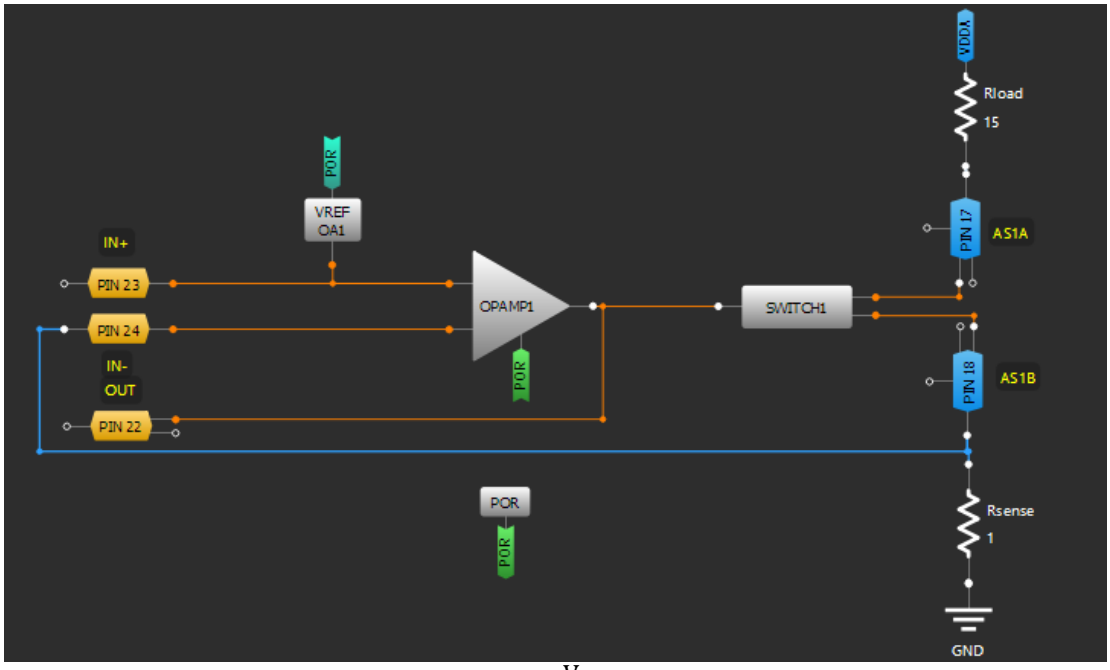
通过特定的连接，运算放大器可以用作电流阱(current sink)。 该电流阱(current sink)通过一个固定检测电阻保持恒定电压。因此，无论负载电阻值如何，流过负载的电流也是恒定的。



所需元器件

- SLG47004V
- 1 个电阻器

GreenPAK 设计图



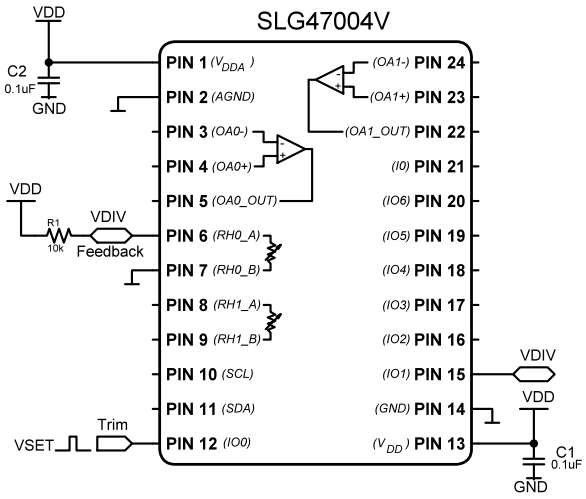
$$I_{load} = \frac{V_{ref}}{R_{sense}}$$

设计步骤

1. 启用 **OpAmp1**，设置带宽(bandwidth selection)为 128 kHz, 并启用电荷泵(Charge Pump 为 Enable CP)，设置 Vref Connection 为 to IN+。
2. 启用 **Vref OA1**，设置 input voltage 为 2.048 V，并设置 output selection 为 96 mV。基准电压决定通过负载的电流。
3. 将 **Switch1** 模式(Mode)设置为 Analog Switch，Big PMOS Control 设置为 by OpAmp。

应用：自动校准

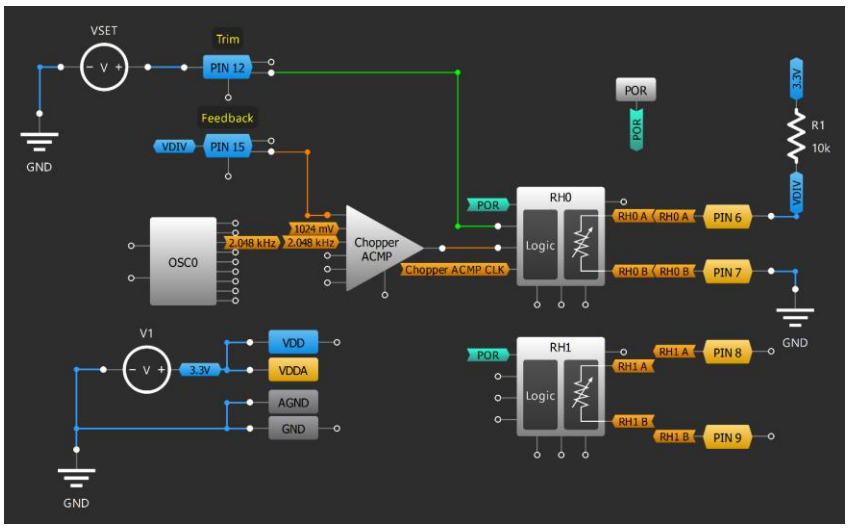
在本实例中, SLG47004 用于实现单点校准的自动校准功能。

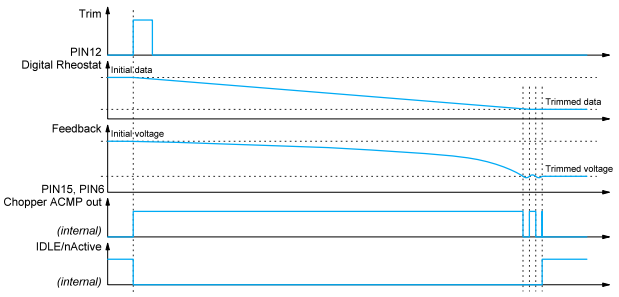


所需元器件

- SLG47004V
- 电阻器

GreenPAK 设计图



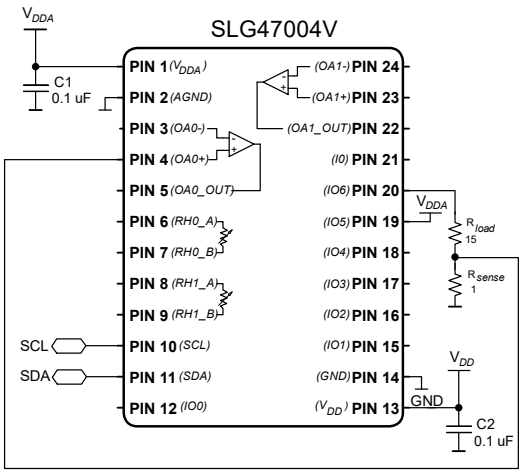


设计步骤

1. 通过 **PIN 12**"Trim"，将 VSET 信号源 脉冲接入 **RH0** 的"SET"端子。
2. 将 **PIN 7** 连接至 GND; 电阻 **R1** (气体传感器、热敏电阻等) 连接至 **PIN 6** 和 **VDD**。将电压反馈 VDIV 自 **PIN 6** 连接至 **PIN 15**。
3. 选择 Channel 0，设置 **Chopper ACMP** 内部参考电压值（IN- CH0 Source）为 1024 mV, CH0 Clock Source 为 OSC0, 以及 IN+ CH0 source 为 external Vref PIN 15。连接 **Chopper ACMP** 输出至 **RH0** 的 nUP/Down
4. 按以下说明配置 **RH0**: Auto-Trim 设置为 Enable, Active level for UP/DOWN 设置为 Up when LOW, Resistance (initial data)设置为 511 (约为 50.7096 kΩ)。

应用：使用运放和 P 沟道 FET 创建电流源

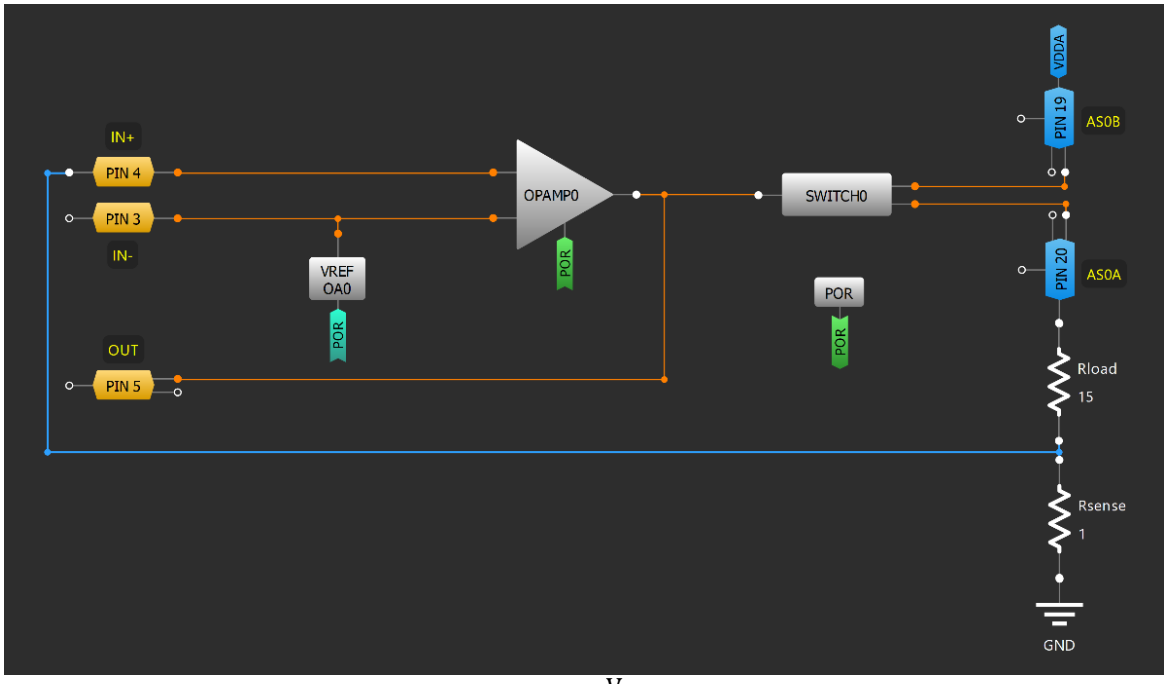
通过特定的连接，运算放大器可以用作电流源。该电流源在恒定检测电阻上保持恒定电压。因此，无论负载电阻值如何，流过负载的电流也是恒定的。



所需元器件

- SLG47004V
- 1 个电阻器

GreenPAK 设计图



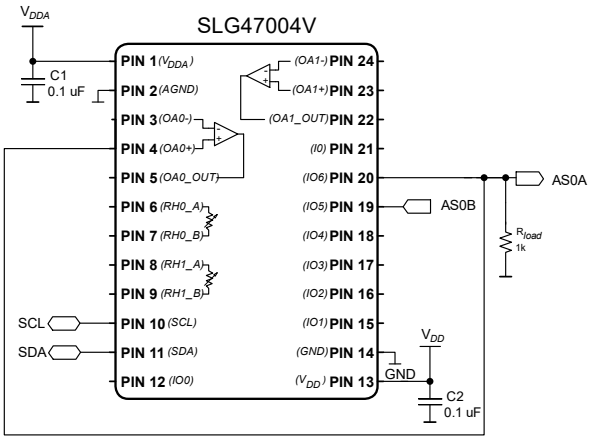
$$I_{load} = \frac{V_{ref}}{R_{sense}}$$

设计步骤

1. 启用 **OpAmp0**，设置带宽(bandwidth selection)为 128 kHz, 并启用电荷泵(Charge Pump 为 Enable CP)，设置 Vref Connection 为 to IN-。
2. 启用 **Vref OA0**，设置 input voltage 为 2.048 V，并设置 output selection 为 96 mV。
3. 将 **Switch0** 模式(Mode)设置为 Analog Switch，Big PMOS Control 设置为 by OpAmp。

应用：使用运算放大器的稳压器

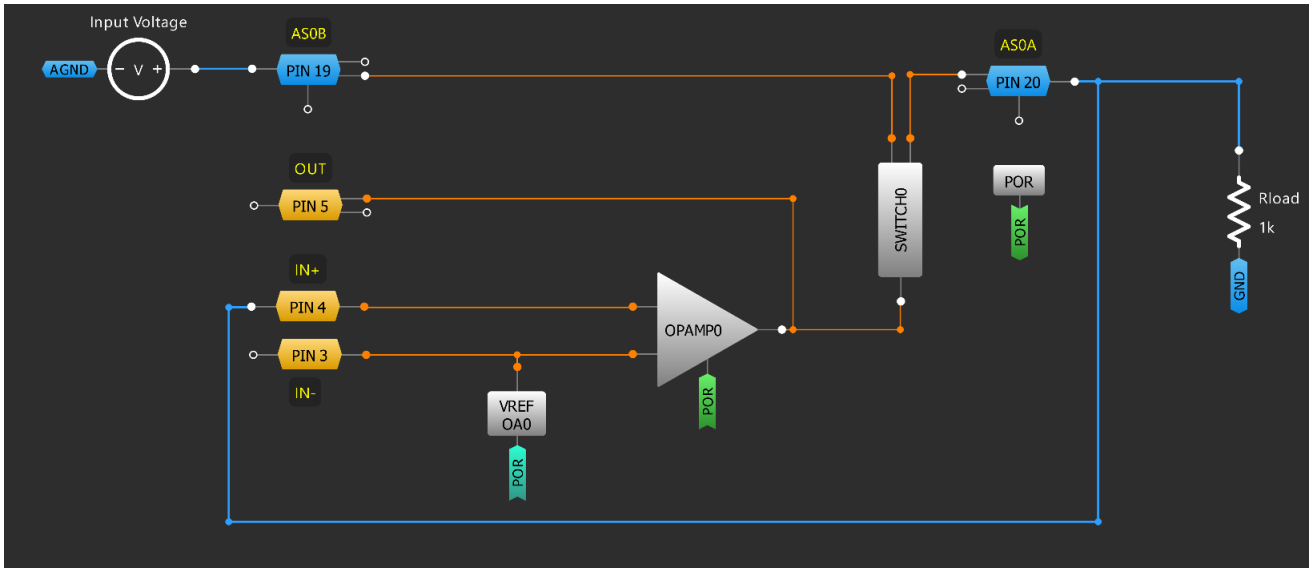
基于运算放大器的稳压器是一种使用运算放大器来调节和稳定输出电压的电路。通过将输出电压与基准电压进行比较，运算放大器调整其输出以保持恒定的电压。这种反馈机制使得稳压器可以根据输入电压的变化来进行调整，从而提供恒定且可靠的输出电压。



所需元器件

- SLG47004V
- 无需其他元件

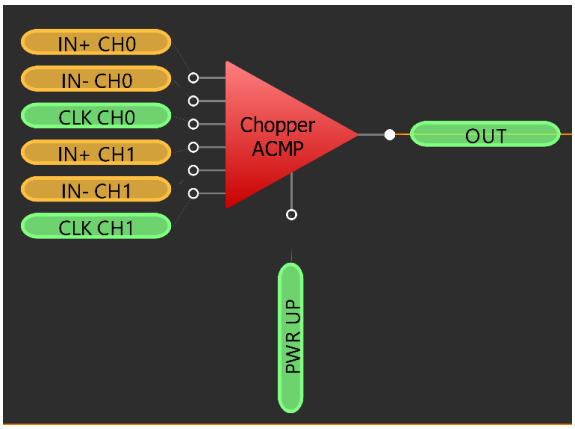
GreenPAK 设计图



设计步骤

1. 启用 **OpAmp0**，设置带宽(bandwidth selection)为 128kHz，并启用电荷泵(Charge Pump 为 Enable CP)，设置 Vref Connection 为 to IN-。
2. 启用 **Vref OA0**，设置 input voltage 为 2.048 V，并设置 output selection 为 1792 mV。
3. 将 **Switch0** 模式(Mode)设置为 Analog Switch，Big PMOS Control 设置为 by OpAmp。
4. 施加输入电压后，无论输入电压如何变化，输出电压都保持稳定并且 $V_{OUT} = V_{REF}$。

技巧：带数字变阻器的斩波比较器



SLG47004 中有一个轨对轨斩波模拟比较器 (Chopper ACMP) 宏单元。可以利用该斩波比较器改变在 Auto-Trim 模式下变阻器(Rheostat)的电阻，进行系统校准（请参阅**应用：自动校准功能**）。

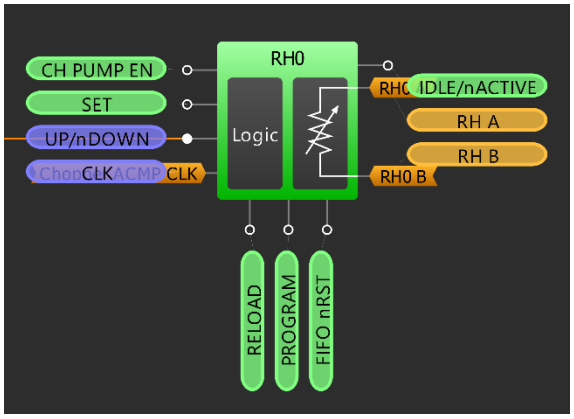
Activation 激活

在 Auto-Trim 模式下，当变阻器正在校准时，自动校准控制逻辑会自动启用斩波比较器。

要使用自动校准功能，必须执行以下预先本配置：

- 在 Digital Rheostat0/1 属性栏中将 Auto-TrIm 设置为 “Enable”。
- 配置 Chopper ACMP 的 IN+ CH0/1，IN+ 作为反馈电压源。如果自动校准功能用于两个变阻器，则必须为两个变阻器都配置 IN+ CH0/1（对于 SET0 被锁存和 SET1 被锁存的情况）。

- 配置 IN- CH0/1，IN- 作为用户设定的阈值电压。如果自动校准功能用于两个变阻器，则必须为两个变阻器配置 IN- CH0/1（对于 Set0 被锁存和 Set1 被锁存的情况）。
- 在 Chopper ACMP 属性中配置 Channel Selection 以与斩波比较器配合使用。



- 设置 Chopper ACMP 的输出属性: inverting 或 non-inverting。
- 选择时钟源 (来自内部 OSC 预分频输出或连接内部其它宏单元输出）。请注意，在 Auto-Trim 模式下，时钟源频率受斩波比较器响应速度的限制。因此，时钟源频率不得大于< f_{ChACMP}> kHz。

- 通过将 RH0/1 模块的 SET0/1 输入设置为高电平来启动自动校准过程。校准过程在 Set 信号的上升沿开始。在自动校准过程结束前，此 Set 信号是锁死的。如果之前未启用 Chopper ACMP 和 Vref，则此 SET 信号将启用它们。计数器开始向上或向下计数，具体取决于 RH0/1（Chopper ACMP 的输出）的 UP/nDOWN 输入脚的电平。如果用户为 Clock 输入选择“Internal Clock”选项，这些时钟脉冲将在校准期间自动生成。时钟脉冲的每个上升沿都会改变计数器的值，从而改变变阻器的值。

如果出现以下三种停止条件之一，自动校准过程就会停止：

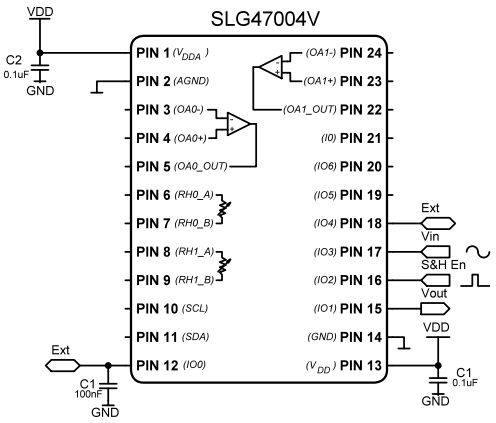
1）在时钟输入上升沿，up/Down 上的信号第二次翻转。
2）变阻器的值达到最大值（1023）。
3）变阻器的值达到最小值（0）。

停止条件导致 IDLE/nACTIVE 信号发生变化，从而重置内部自动校准逻辑。

需要注意的是 SET 信号是边沿敏感的，但如果用户在达到设定点后在此输入脚保持高逻辑电平，可编程校准模块将继续运行并继续围绕设定点调整变阻器。

- 要开始一次新的自动校准过程，用户需要重新将 SET 信号设置为高电平。

应用：取样保持电路

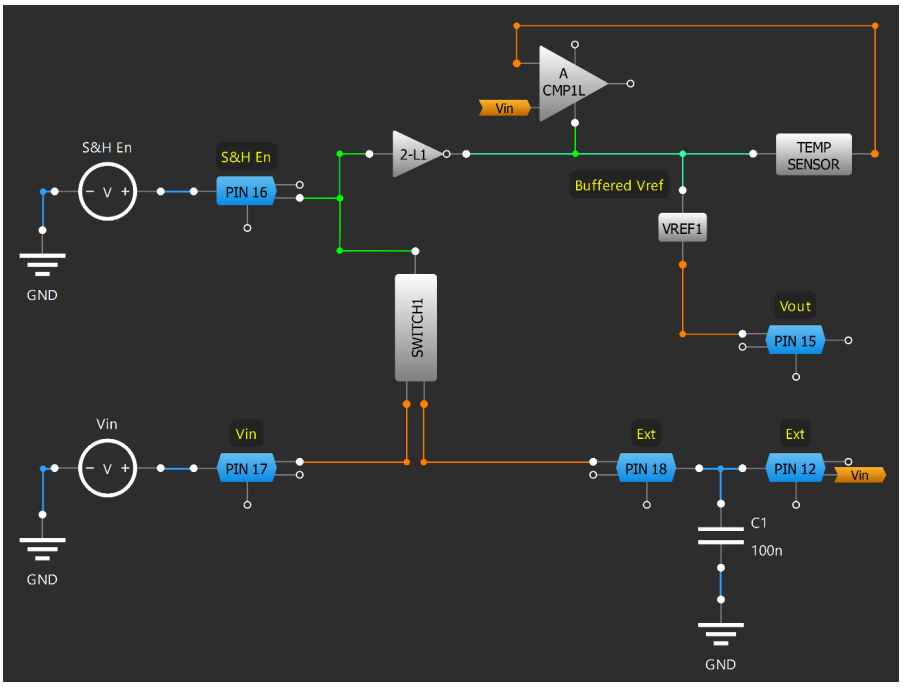


在本应用中，SLG47004 被配置为由模拟开关、电容和缓冲器组成的取样维持电路。

所需元器件

- SLG47004V
- 电容

GreenPAK 设计图



设计步骤

1. 配置 Small PMOS enable 为“Enable by Matrix”，使能 **SWITCH1** 的小型 PMOS。
2. 设置 S&H En 信号并将其连接到 **PIN16** (数字输入，下拉 1M(Digital input, 1M Pull Down))。
3. 将 **SWITCH1 的 ENABLE 端子**连接 **PIN16。**
4. 将 **PIN17** (模拟输入/输出，浮动(Analog input/output, Floating))连接到 Vin 信号源，将 **PIN18** 连接到 **PIN 12**(模拟输入/输出，浮动(Analog input/output, Floating))，并提供与 100nf 电容器的连接。
5. 配置 **2-L1** 作为反相器(Inverter)，并将其 IN0 连接到 **PIN16**，将输出连接到 **ACMP1L** 和 **TEMP SENSOR** 的 PWR UP 端子。
6. 配置 **ACMP1L**: Vref source selection -> VDDA, IN+ source -> TEMP SENSOR output，IN -> Low to High/High to Low source –> Ext. Vref PIN12。
7. 将 **TEMP SENSOR** 的 power down source 配置为“From matrix”。
8. **PIN15** 是取样维持电路的输出(Vout)。