SPC1169 PLL 使用指南

版本 A/0 – 2023 年 4 月

概述

在实际应用中，有时需要确定某个时钟的具体数值，需要将时钟引到芯片管脚上，以便使用示波器进行测量，而通常使用中，用户对 PLL 的频率由来及测量较为感兴趣。本文将描述 PLL 的频率产生过程及将 PLL 时钟引到管脚上进行测量的具体方法。

目录

图片列表

表格列表

版本历史

版本	日期	作者	状态	变更
A/0	2023 年 4 月 23 日	CanChai	Released	首次发布。

术语或缩写

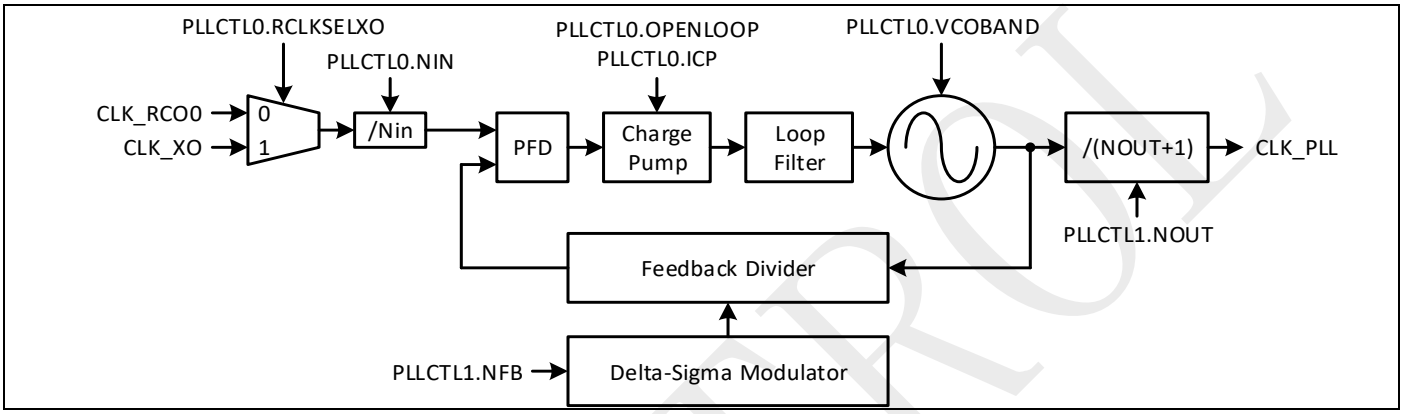
术语或缩写	描述

1 PLL 时钟的产生过程

PLL 模块的整体结构图如图 1-1：PLL 结构图所示，从 RCO 或晶体振荡器作为输入的参考时钟，当环路被锁定时，VCO 频率将在 400MHz 到 600MHz 之间，最后通过可编程分频器进行分频，以提供通常为 25MHz 到 100MHz 的时钟，输出频率由以下公式给出：

$$f_{PLL} = \frac{f_{REF}}{NIN} \times \frac{NFB}{65536} \times \frac{1}{NOUT + 1}$$

图 1-1：PLL 结构图



想要得到某个 PLL 频率，需要按照如下步骤进行计算：

- 选择 NIN 值：PLL 的设计中规定其参考时钟的频率范围为 4MHz~56MHz，而输入到 PFD 子单元的时钟频率应该为 4MHz~8MHz，因此根据不同的参考时钟频率，我们需要设置不同的 NIN 数值，以便得到合理的f_{PFD}数值，手册提供了不同参考频率时需要设置的 NIN 数值，如表 1-1：参考时钟分频设置所示，以便用户快速查阅。
- 设置 NOUT 值：从f_{PLL}计算公式中可以看出，影响最终f_{PLL}的变量过多，不利于计算，所以需要换一个思路来思考怎样计算f_{PLL}的最终值。上文提到环路被锁定时，VCO 的频率将在 400MHz~600MHz，由于 NFB 在反馈环路里，比较难确定，所以先确定 NOUT 的数值，如表 1-2：输出时钟分频设置所示，列出了对应不同输出频率时需要设置的 NOUT 数值。
- 设置 NFB 值：在已经确定 NIN 及 NOUT 数值的情况下，可以推导出 NFB 的计算公式为：

$$NFB = \frac{f_{PLL} \times (NOUT + 1)}{f_{REF} / NIN} \times 65536$$

由于 NFB 的计算公式存在除法运算，这将难免导致最终设置到寄存器中的数据会有部分损失，所以在实际的代码计算过程中，可以根绝最终 PLL 的值对公式做一些变化，避免精度损失，例如：目标 PLL 频率为 100MHz，NIN=4，NOUT + 1=6，则：

$$NFB = \frac{f_{PLL} \times (NOUT + 1)}{f_{REF} / NIN} \times 65536 = \frac{100000000 \times 6}{32000000/4} \times 65536 = \frac{100000000}{25} \times \frac{128}{625} \times 6$$

表 1-1：参考时钟分频设置

参考时钟 (MHz)	参考时钟分频 (NIN = f_{REF}/f_{PFD})
4~8	1
8~16	2
16~24	3
24~32	4
32~40	5
40~48	6
48~56	7

表 1-2：输出时钟分频设置

Desired Output Frequency			Dividing Ratio (NOUT+1)
75MHz	~	100MHz	6
50MHz	~	75MHz	8
37.5MHz	~	50MHz	12
25MHz	~	37.5MHz	16

2 PLL 时钟测量

要想通过示波器测量 PLL 的时钟，那就需要将 PLL 的信号接到芯片 GPIO 上，这里需要注意一个问题，由于 GPIO 的能力限制，所以此时的 PLL 频率不能过高，需要对其进行分频处理，将其设置到合理的频率区间，然后再使用示波器进行测量。

Example Code

```
#include "spd1179.h"
#include <stdio.h>

int main(void)
{
    CLOCK_InitWithRCO(CLOCK_CPU_100MHZ);

    Delay_Init();

    /*
     * Init the UART
     */
    PIN_SetChannel(PIN_GPIO10, PIN_GPIO10_UART0_TXD);
    PIN_SetChannel(PIN_GPIO11, PIN_GPIO11_UART0_RXD);
    UART_Init(UART0, 38400);

    printf("Enter the test\n");

    /* Set PIN_GPIO6/PIN_GPIO7 as GPIO FUNC */
    PIN_SetChannel(PIN_GPIO6, PIN_GPIO6_GPIO6);
    PIN_SetChannel(PIN_GPIO7, PIN_GPIO7_GPIO7);

    /* Set PIN_GPIO6/PIN_GPIO7 direction */
    GPIO_SetPinDir(PIN_GPIO6, GPIO_OUTPUT);
    GPIO_SetPinDir(PIN_GPIO7, GPIO_INPUT);

    PIN_SetChannel(PIN_GPIO2, PIN_GPIO2_CLKMON);
    /* 7:128 div */
    WRITE_FIELD(CLOCK->CLKDETCTL,CLKDETCTL_DCLKDIV_Msk,CLKDETCTL_DCLKDIV_Pos, 7);
    /* 0:RCO0, 1: RCO1, 2: XO, 3: PLL*/
    WRITE_FIELD(CLOCK->CLKDETCTL,CLKDETCTL_DCLKSEL_Msk,CLKDETCTL_DCLKSEL_Pos, 3);
    CLOCK_EnableDetection();

    while (1)
    {
    }
}
```