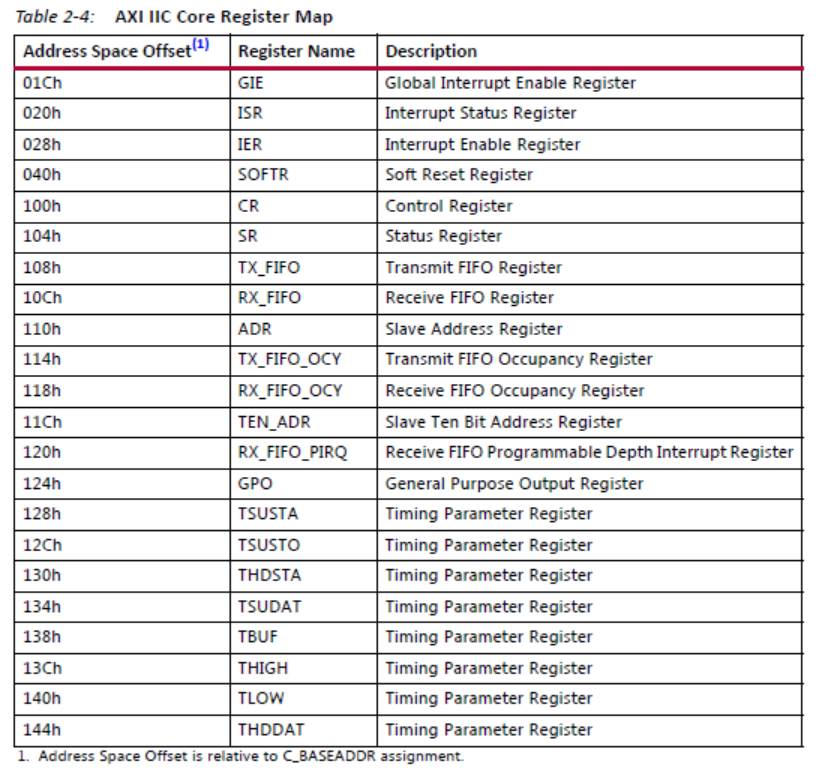
HC-B2010(DIO)卡寄存器定义

版本记录

|  |  |  |  |
| --- | --- | --- | --- |
| FPGA版本 | 修改事项 | 修改时间 | 修改人 |
| 1.00 | 初始版本 | 2020-05-25 | 王勇 |
| 1.03 | 1、修改DO\_deadzone寄存器，可同时设置上升沿和下降沿的死区时间控制  2、PWM捕获和PWM输出使用100MHz时钟  3、修改PWM捕获流程，删除参考时钟设置步骤，修改频率计算时T0参数的取值为10  4、Bit流主时钟修改为62.5MHz  5、Bit流DMA空间大小修改为16KB | 2020-07-08 | 王勇 |
| 1.05 | 1、PWM捕获时钟更改为系统自动判断，无需手动设置。低频时钟由20MHz更改为6.25MHz  2、增加PPS秒脉冲功能  3、增加在线升级相关寄存器  4、修改PWM OUT低频时钟为25MHz | 2020-07-18 | 王勇 |
| 1.08 | 1、PWM捕获主时钟更改为125MHz，低频时钟为125MHz/64  2、修改PWM捕获时间单位及位数  3、修改PWM捕获计数值位数  4、PWM输出主时钟更改为125MHz，低频时钟为125MHz/4  5、修改死区控制时间单位为8ns，范围为8ns~2048ns  6、修改bit流主时钟为125MHz  7、增加bit流输出环形缓冲相关寄存器，并更新bit流输出流程，见2.4  8、修改pps秒脉冲纳秒计数主时钟为125MHz，并修改纳秒的计算过程，见2.8  9、增加125MHz系统时钟校准机制 | 2020-07-30 | 王勇 |

# 寄存器定义

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **名称** | **说明** | **地址(Hex)** | **类型** | **长度(bit)** | **复位值** | **描述** |
| **通用寄存器** | | | | | | |
| Slot | 槽位号 | 000 | R | 5 |  |  |
| Datetime | 程序日期 | 004 | R | 32 |  |  |
| Version | 程序版本 | 008 | R | 32 |  |  |
| Rw\_test | 读写测试 | 400 | RW | 32 | 0x1234ABCD |  |
| **DO相关寄存器** | | | | | | |
| DO\_en | 输出使能 | 480 | RW | 24 | 0 |  |
| DO\_level | 输出电平 | 484 | RW | 24 | 0 |  |
| DO\_pull\_push1 | 输出阻抗模式 | 48C | RW | 32 | 0 | 0~15通道输出阻抗模式，每两个bit代表一个通道，bit[1:0]通道0，bit[3:2]通道1，…，bit[31:30]通道15  Bit值含义：  00：高阻  01：PULL（上拉）  10：PUSH（下拉）  11：PULL&PUSH |
| DO\_pull\_push2 | 输出阻抗模式 | 490 | RW | 16 | 0 | 16~23通道输出模式  ，每两个bit代表一个通道，bit[1:0]通道16，bit[3:2]通道17，…，bit[15:14]通道24  Bit值含义：  00：高阻  01：PULL（上拉）  10：PUSH（下拉）  11：PULL&PUSH |
| DO\_mode1 | 输出模式 | 494 | RW | 32 | 0 | 0~15通道输出模式，每两个bit代表一个通道，bit[1:0]通道0，bit[3:2]通道1，…，bit[31:30]通道15  Bit值含义：  00：无输出  01：电平  10：PWM  11：Bit流 |
| DO\_mode2 | 输出模式 | 498 | RW | 16 | 0 | 16~23通道输出模式，每两个bit代表一个通道，bit[1:0]通道16，bit[3:2]通道17，…，bit[16:15]通道24  Bit值含义：  00：无输出  01：电平  10：PWM  11：Bit流 |
| DO\_pwm\_clksel | PWM输出时钟源 | 49C | R | 24 | 0 | PWM输出时钟，每个bit对应一个通道，bit[0]通道0，bit[1]通道1，…，bit[23]通道23  Bit值含义：  0:125MHz（适合于高频PWM输出）  1:125MHz/4（适合于低频PWM输出） |
| DO\_deadzone | 死区控制 | 4A0 | RW | 3 | 16 | DO输出死区控制，仅用于调试用途。单位10ns，实际的死区控制时间为(deadzone + 1）\*8ns  Bit位定义：  Bit[7:0]:下降沿死区时间控制  Bit[15:8]:上升沿死区时间控制 |
| DO\_ref\_vol0 | 输出参考电压 | 4A4 | RW | 2 | 0 | 通道0~7输出参考电压  Bit[1:0]定义：  0X:5V；  10:12V；  11：外部参考 |
| DO\_ref\_vol1 | 输出参考电压 | 4A8 | RW | 2 | 0 | 通道8~15输出参考电压  Bit[1:0]定义：  0X:5V；  10:12V；  11：外部参考 |
| DO\_ref\_vol2 | 输出参考电压 | 4AC | RW | 2 | 0 | 通道16~24输出参考电压  Bit[1:0]定义：  0X:5V；  10:12V；  11：外部参考 |
| DO\_over\_vol | 输出过流保护/清除过流保护 | 4B0 | RW | 3 | 7 | 输出过流保护状态（读取时）/清除过流保护（写入时），每个bit代表8个通道，bit[0]通道0~7，bit[1]通道8~15,bit[2]通道16~23  读操作时，Bit值含义：  0：正常  1：过流保护  写操作时，Bit值含义：  0：忽略  1：清除过流保护  【注】：  1、过流保护状态会保持，即使过流信号撤销后，该状态也会保持，除非向本寄存器对应bit位写1。  2、复位时，过流信号检测芯片PCA9555的中断引脚默认输出中断，需要在上位机对PCA9555进行读取操作以清除PCA9555的初始中断状态。 |
| DO\_over\_vol\_off | 关闭中断保护 | 4B4 | RW | 3 | 0 | 关闭过流保护功能，仅用于调试用途 |
| PCA\_int\_status | PCA9555中断状态 | 4B8 | R | 3 | 7 | PCA9555中断引脚原始状态，仅用于调试用途。  Bit位含义：  0：无中断  1：有中断  【注】：PCA9555上电复位时，中断引脚默认有输出，需要在上位机对PCA9555进行读取操作以清除初始中断状态。 |
| DO\_over\_vol\_ch | 过流通道 | 4BC | R | 24 | 0 | 过流保护的具体通道号，每个bit代表一个通道 |
| PWM\_out\_hi0 | PWM\_OUT高电平计数 | 500 | RW | 32 |  | 输出通道0，PWM高电平周期计数，单位10ns，实际高电平脉宽为(pwm\_out\_hi +1)\*8ns |
| …… |  |  |  |  |  |  |
| PWM\_out\_hi23 | PWM\_OUT高电平计数 | 55C | RW | 32 |  | 输出通道23，PWM高电平周期计数，单位5ns，实际高电平脉宽为(pwm\_out\_hi +1)\*8ns |
| PWM\_out\_lo0 | PWM\_OUT低电平计数 | 580 | RW | 32 |  | 输出通道0，PWM低电平周期计数，单位5ns，实际低电平脉宽为(pwm\_out\_lo +1)\*8ns |
| …… |  |  |  |  |  |  |
| PWM\_out\_lo23 | PWM\_OUT高电平计数 | 5DC | RW | 32 |  | 输出通道23，PWM低电平周期计数，单位5ns，实际高电平脉宽为(pwm\_out\_lo +1)\*8ns |
| Bits\_clk\_counter | 比特流频率计数器 | 694 | RW | 32 | 99 | 比特流频率计数器，bit流采样率为：  125MHz/(clk\_counter + 1) |
| Bit\_fifo\_head | 比特流环形缓冲头指针 | 6C0 | RW | 32 | 0 | 缓冲区头指针  Bit[31:0]: (512K)  0 ~ 0x7FFFF  【注】：通道1DMA地址偏移：0x0 |
| Bit\_fifo\_tail | 比特流环形缓冲尾指针 | 6C4 | R | 32 | 0 | 缓冲区尾指针。  Bit[31:0]: (512K)  0 ~ 0x7FFFF  【注】：当从DDR读取数据后，应当修改该指针，释放空间。当超过最大地址时，重新从0开始。 |
| Bit\_fifo\_full | 比特流环形缓冲满标记 | 6C8 | R | 1 | 0 | 缓冲区满标记  Bit[0]含义：  0：DDR环形缓冲区未满；  1：DDR环形缓冲区已满； |
| **DI相关寄存器** | | | | | | |
| DI\_in\_level | DI电平信号 | 880 | R | 32 | 0 | DI电平输入信号，每个通道代表一个 |
| PWM\_cap\_en | PWM捕获使能 | 884 | RW | 32 | 0 | PWM捕获使能，每个bit代表一个通道使能 |
| PWM\_cap\_clksel | PWM捕获参考时钟 | 888 | R | 32 | 0 | PWM捕获参考时钟，每个bit代表一个通道的时间单位。  Bit位定义：  0: 125MHz  1: 125MHz/64  【注】该参考时钟为系统根据捕获结果自动调整，确保相关通道的电平计数和周期计数不会溢出 |
| PWM\_cap\_done | PWM捕获完成状态 | 88C | RW | 32 | 0 | 在设置的捕获时长内，完成了捕获，每个bit代表一个通道 |
| PWM\_cap\_duration0 | PWM捕获时长 | 900 | RW | 17 | 0 | 通道0 PWM捕获时间计数（单位单位131.072us） |
| …… |  |  |  |  |  |  |
| PWM\_cap\_duration31 | PWM捕获时长 | 97C | RW | 17 | 0 | 通道31 PWM捕获时间长度（单位单位131.072us） |
| PWM\_cap\_hi0 | PWM高电平累积计数 | 980 | RW | 28 |  | 通道0 PWM捕获高电平计数（单位：8ns） |
| …… |  |  |  |  |  |  |
| PWM\_cap\_hi31 | PWM高电平累积计数 | 9FC | RW | 28 |  | 通道31 PWM捕获高电平计数（单位：8ns） |
| PWM\_cap\_period0 | PWM周期累积计数 | A00 | RW | 28 |  | 通道0 PWM捕获高、低电平整周期计数（单位：8ns） |
| …… |  |  |  |  |  |  |
| PWM\_cap\_period31 | PWM周期累积计数 | A7F | RW | 28 |  | 通道31 PWM捕获高、低电平整周期计数（单位：8ns） |
| PWM\_cap\_cycles0 | PWM周期数 | A80 | RW | 9 |  | 通道0 PWM捕获周期数 |
| …… |  |  |  |  |  |  |
| PWM\_cap\_cycles31 | PWM周期数 | AFC | RW | 9 |  | 通道31 PWM捕获周期数 |
| **PPS相关寄存器** | | | | | | |
| Pps\_source | PPS源 | C00 | RW | 1 | 0 | PPS源  Bit[0]含义：  0：PXI Trigger  1：内部PPS |
| Pps\_edge\_sel | PPS边沿选择 | C04 | RW | 1 | 0 | PPS边沿选择  Bit[0]含义:  0:下降沿有效  1:上升沿有效 |
| Pps\_trig\_sel | PXI TRIGGER选择 | C08 | RW | 3 | 0 | PXI TRIGGER选择  Bit[2:0]含义：PXI Trigger 0~7 |
| Pps\_second\_set | 系统秒计数设置 | C0C | RW | 32 | 0 | 系统秒计数设置  Bit[31:0]含义：32位的秒计数值 |
| Pps\_second\_valid | 系统秒计数设置有效 | C10 | W | 1 | 0 | 系统秒计数设置有效  Bit[0]含义：  0：忽略  1：设置秒计数值有效 |
| Pps\_export\_en | PPS输出使能 | C14 | RW | 8 | 0 | PPS输出使能  Bit[7:0]含义：  每个bit代表一跟PXI Trigger线，0代表不输出PPS，1代表将内部PPS从PXI Trigger线输出 |
| Pps\_seconds | 系统秒计数 | C18 | R | 32 | 0 | 系统秒计数  Bit[31:0]含义：系统的秒计数绝对值  【注】用户设置秒计数会对系统秒计数值重新初始化；每一个PPS秒脉冲到来时，秒计数值加1； |
| Pps\_nano\_counter | 系统纳秒计数 | C1C | R | 32 | 0 | 系统纳秒计数  Bit[31:0]含义：系统的纳秒计数，单位8ns。  【注】  1、该值的最大值取决于125MHz时钟校准计数值  2、该值始终小于125MHz时钟校准计数值，如果没有秒脉冲，该值将保持最大值不变。每次秒脉冲到来会重新复位该值。 |
| Pps\_counter | PPS秒计数 | C20 | R | 32 | 0 | PPS秒计数  Bit[31:0]含义：接收到的PPS秒脉冲个数。 |
| Bits\_seconds | BITSTREAM发送时间秒计数 | 69C | R | 32 | 0 | Bit[31:0]含义：发送Bit流第一个采样点的秒计数绝对值  【注】当Bit流输出功能从禁止更改为使能后，该值将在下一次Bit流发送的第一个采样点时刻被更新 |
| Bits\_nano\_seconds | BITSTREAM发送时间纳秒计数 | 6A0 | R | 32 | 0 | Bit[31:0]含义：发送Bit流第一个采样点的纳秒计数绝对值  【注】当Bit流输出功能从禁止更改为使能后，该值将在下一次Bit流发送的第一个采样点时刻被更新 |
| **系统时钟校准** | | | | | | |
| Cal\_counter | 系统时钟校准计数 | C40 | R | 32 | 0 | 系统时钟校准计数，该计数是1秒内125MHz主时钟的计数值 |
| Cal\_done | 系统时钟校准完成 | C44 | R | 1 | 0 | 系统时钟校准完成，0为未校准完成；1标识校准完成 |
| **EEPROM IIC操作寄存器（校准数据存取）** | | | | | | |
| 地址偏移：0x2000，详细定义参考XILINX pg090-axi-iic.pdf，table2-4  EEPROM IIC操作方法，参考复旦微FMC24C64D数据手册。 | | | | | | |
| **DAC IIC操作寄存器（配置输入参考电压）** | | | | | | |
| 地址偏移：0x4000，详细定义参考XILINX pg090-axi-iic.pdf，table2-4  DAC IIC操作方法，参考Microchip MCP4728数据手册。 | | | | | | |
| **PCA9555 IIC操作寄存器（读取过流保护状态）** | | | | | | |
| 地址偏移：0x6000，详细定义参考XILINX pg090-axi-iic.pdf，table2-4  PCA9555 IIC操作方法，参考NXP PCA9555数据手册。  **操作伪代码：**  1、初始化RX\_FIFO\_PIRQ  Write 0x6120 0xf  2、复位TX\_FIFO  Write 0x6100 0x2  3、使能IIC、撤销TX\_FIFO复位,禁止General Call  Write 0x6100 0x1  4、发送器件地址+START+WRITE  Write 0x6108 0x140（3片的PCA9555地址不一样，需要改为实际的地址）  5、发送COMMAND BYTE  Write 0x6108 0(低边8位command为0，高边8位command为1)  6、发送器件地址+START+READ  Write 0x6108 0x141（3片的PCA9555地址不一样，需要改为实际的地址）  7、发送STOP  Write 0x6108 0x200  8、检查RX\_FIFO是否为空(2 : BB; 3: SRW; 4 : TX\_FIFO\_FULL; 5 : RX\_FIFO\_FULL; 6 : TX\_FIFO\_EMPTY; 7 : RX\_FIFO\_EMPTY)  Read 0x6104  9、读取RX\_FIFO  Read 0x610C | | | | | | |
| **在线升级相关寄存器** | | | | | | |
| 地址偏移：0xA000，详细定义参考XILINX pg153-axi-quad-spi.pdf。  Flash的读写操作参考S25FL256SAGNFI00配置芯片数据手册。 | | | | | | |



# 常见功能设置流程

## 时钟校准

系统125MHz主时钟存在固定的频偏，需要进行校准。

读取Cal\_counter寄存器，时钟校准系数***clk\_coef*** = 125000000/ Cal\_counter。如果该值＞1，表明实际的时钟频率＞125MHz；反之，如果该值＜1，则表明实际的系统时钟＜125MHz；

## 电平输入

操作步骤：

1. 配置MCP4728 DAC寄存器，设置输入比较参考电压，该值可以持久保存。
2. 读取DI\_in\_level(0x880)寄存器，bit为为1代表输入为高电平，bit位为0，代表低电平输入。

对于PWM输入信号，虽然也可以读取电平，但是电平值在变化，故实际读取的结果无效。

## 电平输出

操作步骤：

1. 设置DO\_en寄存器，通道对应bit位设置为0，禁止输出；
2. 设置DO\_level寄存器，设置输出电平，每个bit代表一个通道；
3. 设置DO\_pull\_push<M>(M=0..1)寄存器,设置输出阻抗模式;
4. 设置DO\_mode<M>(M=0..1)寄存器，设置通道输出模式为01（电平模式）；
5. 设置DO\_en寄存器，通道对应bit位设置为1，使能输出；

## PWM输出

操作步骤：

1. 设置DO\_en寄存器，通道对应bit位设置为0，禁止输出；
2. 设置DO\_pwm\_clksel寄存器，设置PWM输出时钟源为125MHz或31.25MHz（输出PWM频率≥10kHz时，时钟源设置为125MHz；＜10kHz时，时钟源设置为31.25MHz），参考时钟频率记为F0(Hz)；
3. 根据待输出的频率Freq(Hz)和占空比Duty（百分比）计算高低电平脉宽,需转换为高低电平计数，分别为：高电平计数***clk\_coef*** \* Duty \* F0/Freq – 1;低电平计数为：clk\_coef \* (1 – Duty) \* F0/Freq。这里应用了时钟校准系数clk\_coef修正输出频率。举例，如果F0 ＞ 125MHz时，clk\_coef ＞ 1。电平计数乘以clk\_coef的结果使得电平计数器值变大，从而减小输出频率，达到修正输出频率的目的；此外，由于高频输出时，一个计数器就是8ns，其带来的误差超过系统时钟本身的误差，故时钟校准的效果不明显；低频时时钟校准的效果明显；
4. 设置PWM\_out\_hi<N>(N=0..23)寄存器，设置高电平脉宽，实际脉宽为(PWM\_out\_hi<N> +1)\*8ns；
5. 设置PWM\_out\_lo<N>(N=0..23)寄存器，设置对应通道低电平脉宽，实际脉宽为(PWM\_out\_lo<N>+1)\*8ns；
6. 设置DO\_pull\_push<M>(M=0..1)寄存器,设置输出阻抗模式;
7. 设置DO\_mode<M>(M=0..1)寄存器，设置通道输出模式为10（PWM OUT模式）；
8. 设置DO\_en寄存器，使能输出；

## Bit流输出

操作步骤：

1. 设置DO\_en寄存器，通道对应bit位设置为0，禁止输出；
2. 设置DO\_pull\_push<M>(M=0..1)寄存器,设置输出阻抗模式;
3. 设置DO\_mode<M>(M=0..1)寄存器，设置通道输出模式为11（BITSTREAM模式）；
4. 设置Bits\_clk\_counter频率计数器寄存器，实际频率Freq = ***clk\_coef*** \* 125MHz/(counter+1)；与PWM输出一样，这里也需要对输出频率应用时钟校准系数clk\_coef进行校准。同理，高频时，时钟校准效果不明显；低频时，时钟校准效果较明显；
5. 设置DO\_en寄存器，使能输出；
6. 读取Bits\_fifo\_full寄存器，如果Bits\_fifo\_full的值为1，则延迟1ms，重复当前步骤；
7. 读取Bits\_fifo\_head、Bits\_fifo\_tail，计算待发送数据的长度：

待发送数据长度为以下各项的最小值：

* 环形缓冲剩余字节数：(Bits\_fifo\_tail - Bits\_fifo\_head + 0x80000) & 0x7FFFF - 16；
* 环形缓冲最大长度：0x80000；
* 当Bits\_fifo\_tail小于Bits\_fifo\_head时，环形缓冲尾部的长度：0x80000 - Bits\_fifo\_head；
* 剩余待发送长度（需8字节对齐）：(剩余长度 + 0x7) & ~0x7；

1. 读取bit流数据，从Bits\_fifo\_head开始，DMA写入数据，长度为上一步计算的结果；
2. 更新Bits\_fifo\_head寄存器，新Bits\_fifo\_head = (旧Bits\_fifo\_head + DMA长度) & 0x7FFFF；
3. 更新已发送bit流文件长度，如果已发送长度≥文件长度，则结束，否则重复步骤6~10；
4. 设置DO\_en寄存器，通道对应bit位设置为0，禁止输出；

## PWM捕获

操作步骤：

1. 设置PWM\_cap\_en寄存器，对应通道设置为0，禁止PWM捕获；
2. 设置PWM\_cap\_duration<N>(N=0..31)寄存器，设置捕获时长。捕获时长的单位是8ns\*16384 = 131072ns，即131.072us；
3. 设置PWM\_cap\_en寄存器，对应通道设置为1，使能PWM捕获；
4. 轮询PWM\_cap\_done寄存器，直至对应通道的bit位为1时，代表捕获完成；
5. 读取PWM\_cap\_hi<N>(N=0..31)寄存器，读取捕获的高电平计数值；
6. 读取PWM\_cap\_period<N>(N=0..31)寄存器，读取捕获的低电平计数值；
7. 读取PWM\_cap\_cycles<N>(N=0..31)寄存器，读取捕获的PWM周期个数（即PWM输入信号上升沿个数）；
8. PWM捕获锁使用的系统时钟为FPGA自动判断。读取PWM\_cap\_clksel寄存器，每一个bit代表一个通道，bit0代表0通道，依次类推；当bit位为0时，捕获时钟频率F0=125MHz;当bit位为1时，捕获时钟频率F0=125MHz/64；
9. 计算信号占空比和频率。信号占空比为PWM\_cap\_hi<N>/PWM\_cap\_period<N>，信号频率为：F0/( ***clk\_coef*** \* PWM\_cap\_period<N> / PWM\_cap\_cycles<N>)；同样，由于使用了125MHz系统时钟，也需要应用时钟校准系数clk\_coef对捕获的时钟进行校准。

## 过流保护

当出现过流保护时，板卡以组（每8路输出为一组）为单位关闭高低边的MOS管输出，可读取DO\_over\_vol寄存器检查是否出现过流保护。

如排故完成，需要重新恢复，只需向DO\_over\_vol寄存器对应组的bit位写1即可。

【注】上电时，如果PCA9555的输出了中断信号，此时需要先通过IIC读取PCA9555的inport端口状态，检查是否全0，如果全为0，表明过流信号已经撤除，此时再向DO\_over\_vol寄存器对应组的bit位写1，即可恢复正常状态。

## PPS秒脉冲

PPS秒脉冲的纳秒计数使用了125MHz系统时钟进行计数，故实际的纳秒数需要进行校准。纳秒数 = Pps\_nano\_counter \* 8 \* ***clk\_coef***。此外，注意纳秒数最大不能超过999999999。