PLL Programming Guide

Application Note Addendum

Document Revision: 1.0.0, Oct 2020

Printed in the United States.

SILICON CREATIONS LLC., 1745 North Brown Rd, Suite 200, Lawrenceville, GA 30043.

For support email support@siliconcr.com

1 Contents

Contents

List of Figures

List of Tables

2 Overview

This document provides guidelines for programming a PLL.

3 PLL Configuration Guidelines

3.1 Configuring a PLL

The Figure 1 shows a typical Integer PLL system and Figure 2 shows a fractional PLL system. The relationship between FOUT* and FREF is shown by the equation below.

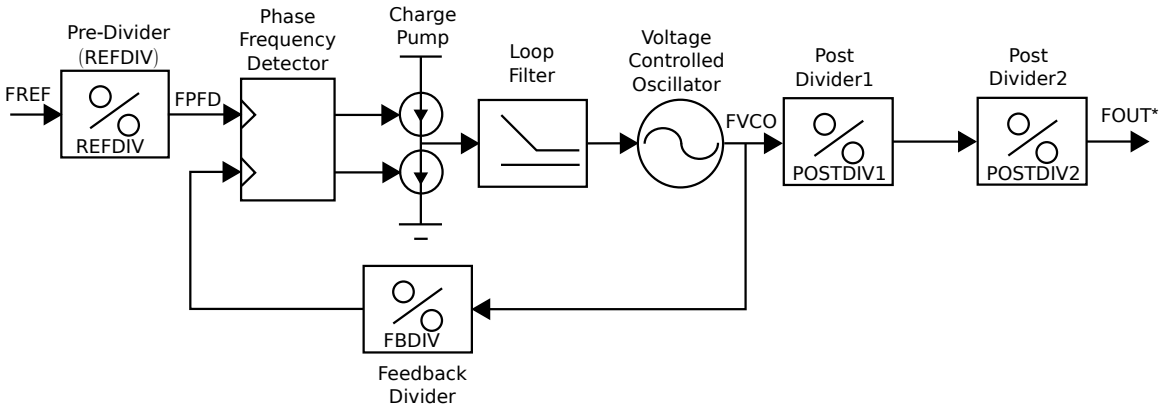

Figure 1: Configuring a PLL in Integer Mode.

$$\text{FOUT*} = \frac{\text{FREF*FBDIV}}{\text{REFDIV*POSTDIV1*POSTDIV2}}$$

This is a generic equation, please refer to the specific datasheet. FOUT is the prefix to PLL output ports. Check the datasheet for relevant pin names.

It is importnat to ensure FBDIV is programmed to a valid setting anytime the PLL is enabled. Some PLLs may map FBDIV=0 to a higher value which may prevent the PLL from LOCKing. ****NOTE**: If a PLL has multiple post dividers in series, the value of the first post divide should be maximized before enabling the second post divide.

For example, Divide-by-4:

- Recommended: Set POSTDIV1=4, POSTDIV2=1
- Not Recommended but acceptable: Set POSTDIV1=2, POSTDIV2=2
- Not Acceptable: POSTDIV1=1, POSTDIV2=4

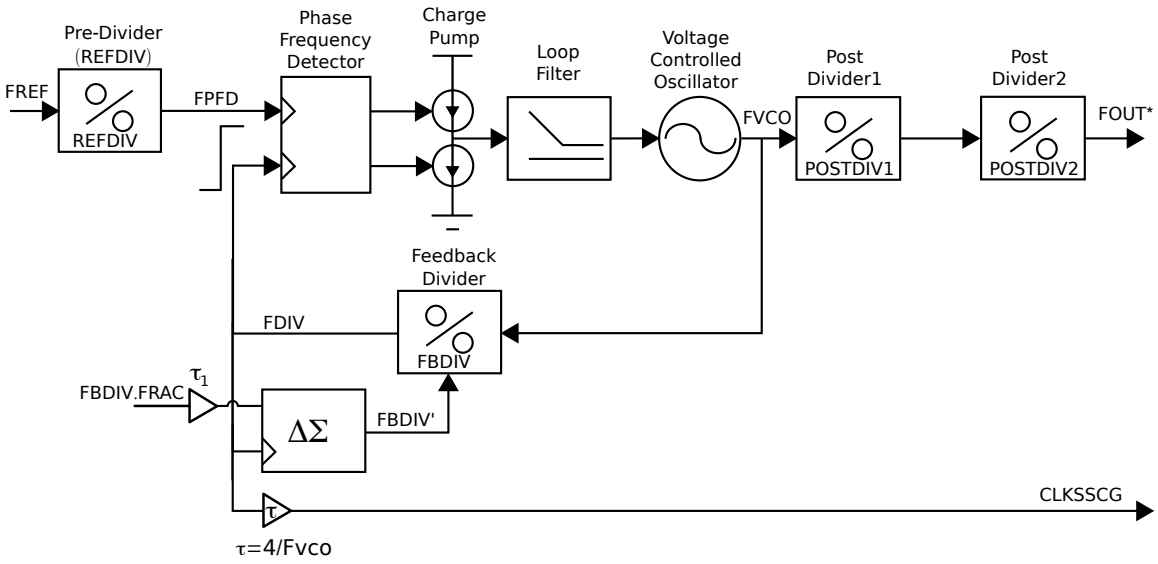

Figure 2: Configuring a PLL in Fractional Mode.

$$\text{FOUT*} = \frac{\text{FREF}}{\text{REFDIV}} \times \frac{\text{FBDIV}+\frac{\text{FRAC}}{2^{24}}}{\text{POSTDIV1*POSTDIV2}}$$

This is a generic equation, please refer to the specific datasheet.

****REMINDER**: If a PLL has multiple post dividers in series, the value of the first post divide should be maximized before enabling the second post divide.

For example, Divide-by-4:

- Recommended: Set POSTDIV1=4, POSTDIV2=1
- Not Recommended but acceptable: Set POSTDIV1=2, POSTDIV2=2
- Not Acceptable: POSTDIV1=1, POSTDIV2=4

3.2 Programming the Feedback Divide

There are three programming conditions for Integer/Fractional Divide values that will be discussed in this section:

- Static divide in integer or fractional mode (no updates)
- Dynamic Integer divide in integer mode
- Dynamic Integer/Fractional divide in fractional mode

For static divide values, no special re-timing logic is required. The integer and fractional (if used) divide values should be programmed while the PLL is disabled (PLLEN=0, EN=0 or PD=1).

Real-time feedback divide updates, such as those discussed in Section 3.3 should not be made asynchronously as a glitch may occur.

Figure 3, below shows a recommended block diagram for PLL connections if the feedback divide is to be changed dynamically while in integer mode.

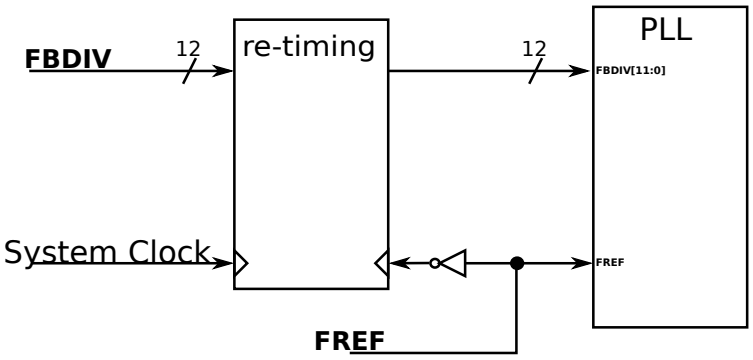

Figure 3: Feedback Divide Update Logic, Integer Mode

In this mode, the falling edge of FREF is used to clock the FBDIV updates to the PLL. Use of this approach is limited by synthesis constraints, and so for faster reference clocks (e.g. >> 150MHz) care should be taken to ensure sufficient timing margin is available. It's important to ensure the FBDIV value ariving at the PLL input remains valid anytime the PLL is enabled.

Figure 4 shows a recommended block diagram for PLL connections if the feedback divide is to be changed dynamically while in fractional mode.

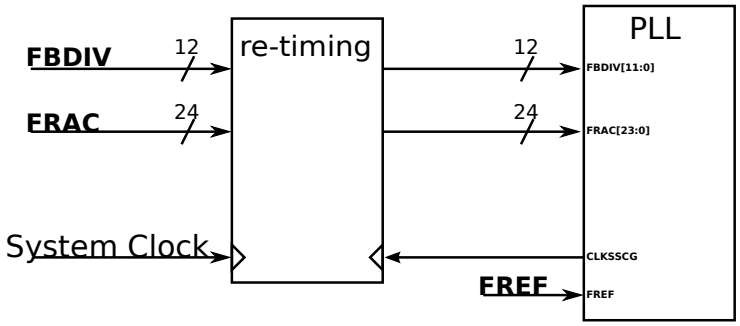

Figure 4: Feedback Divide Update Logic, Fractional Mode

In fractional mode, updates to the feedback divide should be made on the rising edge of CLKSSCG, which is an output clock from the PLL. CLKSSCG is only active while the PLL is operating in fractional mode (DSMPD=0 or DSMEN=1). For information about timing constraints on CLKSSCG, see Section 3.2.1, below.

3.2.1 CLKSSCG Synthesis Constraints

- CLKSSCG is a low duty cycle output clock from the PLL
- CLKSSCG is active while the PLL is in in fractional mode (DSMPD=0 or DSMEN=1) and is enabled (PD=0 or PLLEN=1).
- The frequency of CLKSSCG is equal to the reference clock frequency (FREF) divided by the reference divide value. This frequency is the same as the PLL PFD frequency, the limits of which are specified in the PLL Datasheet.
- The pulse width of CLKSSCG is relative to a fixed number of VCO periods that varies depending on the PLL architecture. This pulse-width is provided in the specifications table of the PLL datasheet.
- For PLLs with a maximum VCO frequency under 5 GHz, the pulse width of CLKSSCG is typically 4 VCO periods, but may be 8 or higher.
- CLKSSCG has no defined timing relative to any of the PLL output clocks.
- Detailed SDC and Liberty Model description is proided in Digital_Integration_Appnote which is included in delivery of an IP.

An example .sdc definition for CLKSSCG is shown in Figure 5, below.

```
### Timescale: 1ns

#Clock Generation
create_clock -name "CLKSSCG" -period 25 -waveform { 0 4 } [get_ports CLKSSCG]
#Clock: CLKSSCG (CLKSSCG), Max Frequency: 40MHz, Worst Case Duty Cycle: 16%

#Clock Uncertainty
set_clock_uncertainty 0.05 [get_clocks CLKSSCG]

#False Path Definitions
set_false_path -from [get_clocks CLKSSCG] -to [get_clocks [remove_from_collection [all_clocks] CLKSSCG]]
```

Figure 5: Example CLKSSCG .sdc

In the example, the reference frequency is 40 MHz, the reference divide is 1, and the feedback divide is 25, resulting a VCO frequency of 1GHz. For this PLL, CLKSSCG has a pulse-width of 4 VCO periods, and so the total high time is 4*1ns=4ns out of a 25ns reference period for a duty cycle of (4*1ns/25ns)=16%. For lower PFD frequencies CLKSSCG duty cycle percentage may be even lower than 16%.

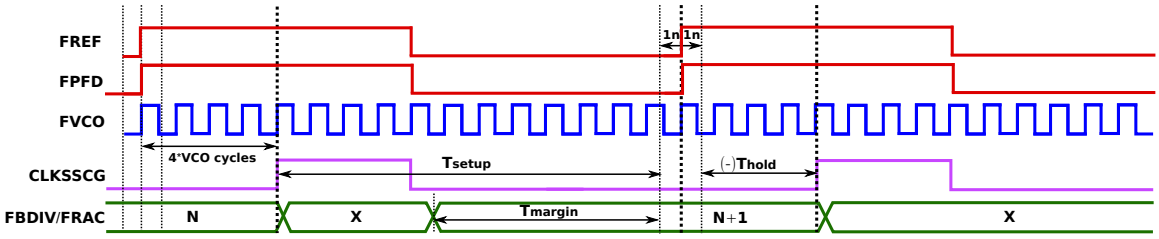

Figure 6: Timing Diagram of CKSSCG

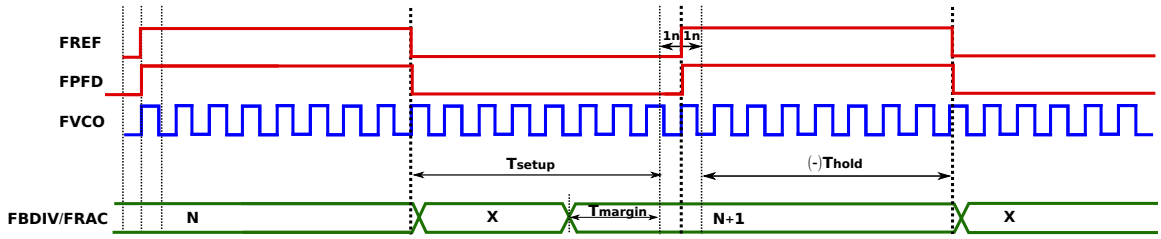

Figure 7: Timing Diagram of FREF

Generic Numerical Example :

- F_{PFD}: 40MHz
- F_{VCO}: 1GHz
- $T_{\text{uncertainty}}$: 1ns
- T_{setup} : $T_{\text{PFD}} - (4 * T_{\text{VCO}}) - 1ns = 25ns - 4n - 1n = 20ns$
- Thold: 0 (actual hold time is negative)

In this particular example, pulse width of CLKSSCG is 4*T$_{VCO}$.

- F_{FREF}: 40MHz
- $DutyCycle_{\text{FREF}}$: 45%-55%
- F_{VCO}: 1GHz
- $T_{\text{uncertainty}}$: 1ns
- T_{setup} : $T_{\text{FREF}} * (DutyCycle_{\text{FREF}}) - 1ns = 25ns * 0.45 - 1n = 10.25ns$
- Thold: 0 (actual hold time is negative)

3.2.2 Synthesis Constraints for PLL's Output Clocks

Following is a simple guideline for synthesis constraints for PLL's output clocks.

- Create_clock for FREF, FOUTVCOinternalclk, FOUTPOSTDIVinternalclk, CLKSSCGinternalclk
- Create_generated_clock for FOUTVCO (divide or multiply 1 of FOUTVCOinternalclk), FOUTPOSTDIV (divide or multiply by 1 of FOUTPOSTDIVinternalclk), CLKSSCG (divide or multiply by 1 of CLKSSCGinternalclk), other FOUT* pins that you are using (divide by the appropriate value of FOUTPOSTDIVinternalclk)
- Refer to Digital Integration Appnote for additional details

Figure 8 below illustrates relationship in the 4-phase divider present in some PLL versions that provides FOUT1PH0/90/180/270.

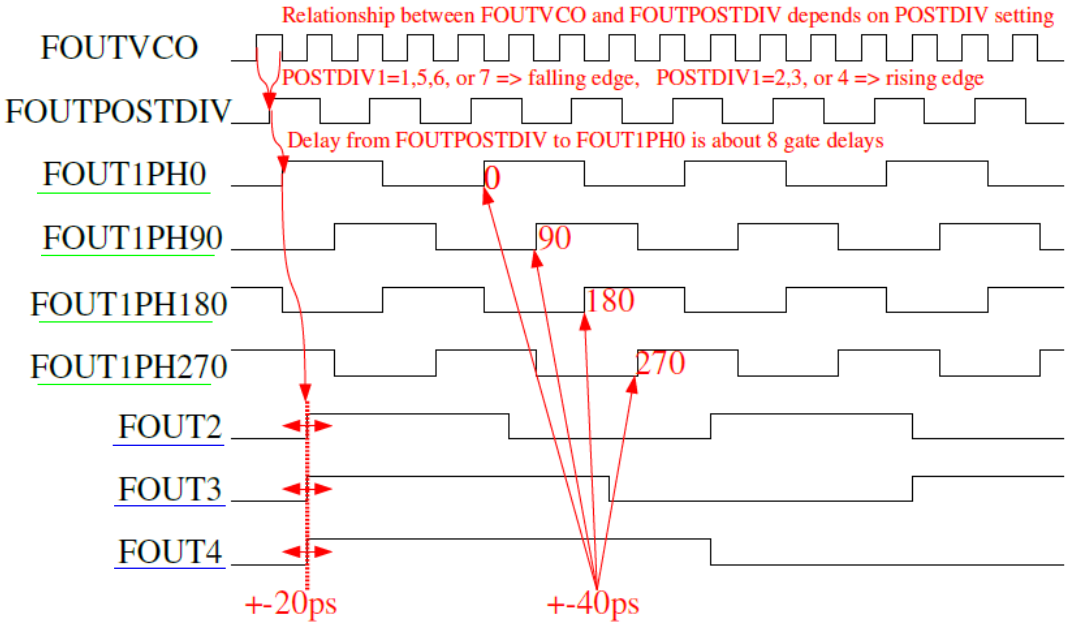

Figure 8: Post divide output waveform timing

3.3 Real-time frequency changes

The frequency of the VCO can be changed while the PLL is running by changing the feedback divide ratio. In many cases, a discrete step in the divide ratio will result in an over-shoot in the output frequency.

If the circuitry clocked by the PLL cannot tolerate an over-shoot, it is recommended to gate the output clock using a clock gating cell. An example of the configuration and the corresponding waveforms are shown in Figure 9. The wait time for EN to go high after FBDIV is updated depends on the specific PLL. Please check the verilog model for the timing information. Alternatively, a very long time, e.g. 256 PFD cycles, is typically sufficient.

Figure 10 shows the instantaneous frequency excursions when frequency is stepped up.

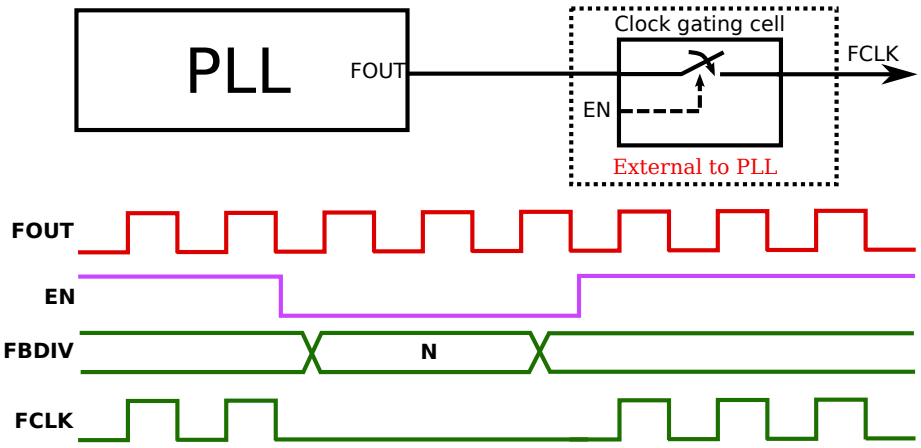

Figure 9: Output Clock Gating Cell

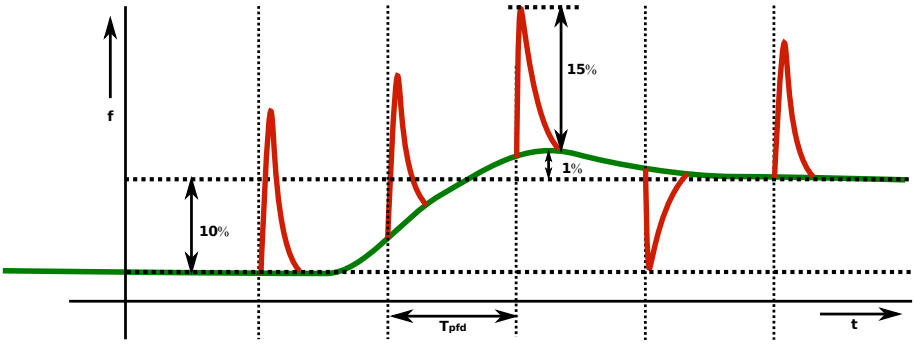

Figure 10: Real-time Frequency Change

If clock gating is not desireable the frequency may be changed over multiple small steps e.g less than 0.1% per PFD clock period. Typically, the size of the frequency ”spikes” are proportional to the PFD’s frequency step. For frequency step of more than 0.01% per PFD clock period, it is recommended to run gate-level verilog simulations.

Spread spectrum modulator typically steps frequency by less than 0.005% per PFD clock period, so does not generate frequency spikes such as in Figure 10.

3.4 Spread spectrum modulator

Spread Spectrum Modulator (SSMOD) is a separate IP provided as synthesizable RTL which works with Phase Locked Loop (PLL) to produce spread spectrum clock generation (SSCG). SSMOD modulates the feedback divider value of the PLL, thus modulating the PLL's output frequency. The modulated clock spreads the energy so that it falls into a large number of the frequency bands without putting enough energy into any one band to exceed the statutory limits.

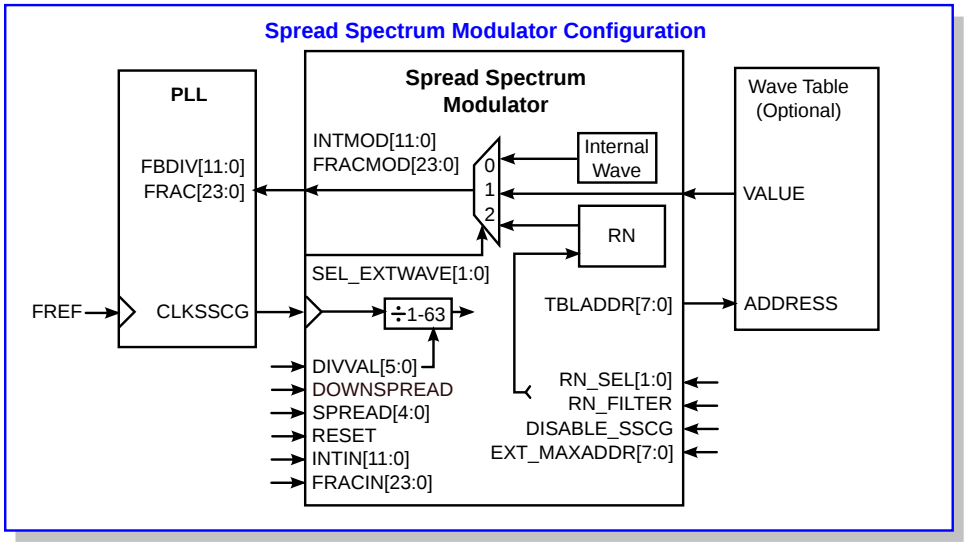

Figure 11: Top-level block diagram

The general configuration is shown in the figure 11. It is important that the PLL be configured in fractional-N mode to preserve the accuracy of modulation. SSMOD may not be used when fixed frequency is required and also if low long-term jitter is of concern.

4 Power-Up Sequence

The following figure shows the power-up sequence for most PLLs.

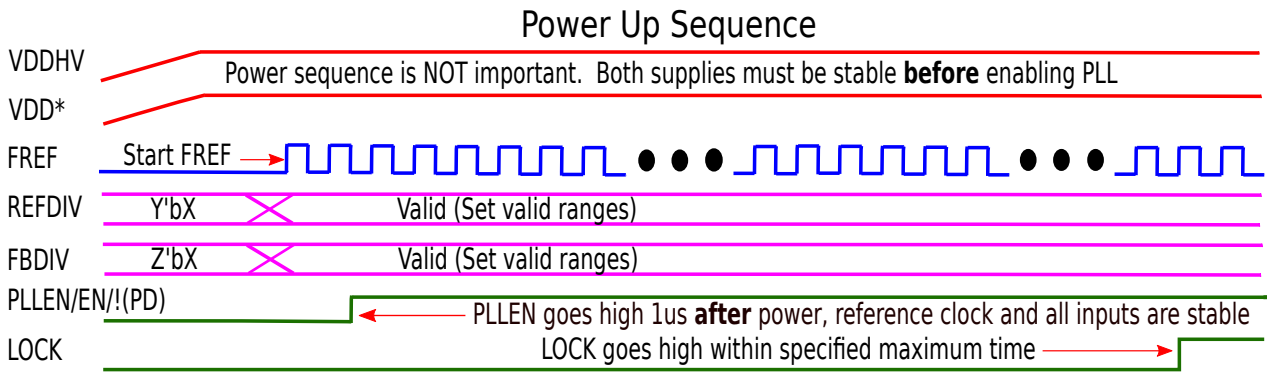

Figure 12: Startup Sequence

- VDDHV/VDD are generic I/O supply and core supply domains.
- All PLL supplies must be powered on.
- FREF is ignored when PLL is powered down but must be stable at the time PLL is enabled.

5 Production Testing

The following tests are recommended for determining functional yield during production screening:

- Measure output clock frequency, and ensure it is within the expected tolerance
  - Recommended to average at least 10,000 output clock periods
    - Tolerance should include some margin for jitter
  - For the frequency measurement, choose a setting which uses the maximum divider values and maximum VCO frequency required for the application
    - All lower divide values are inherently verified when using the higher divide setting
  - Measure the LOCK signal and ensure it goes high within the specified amount of time
  - For PLLs containing OFFSET calibration, OFFSETCALOUT may be sampled to determine mismatch outliers
  - OFFSETCAL remaining low after specified LOCK time may indicate functional failure

6 HTOL testing

Use the maximum VCO frequency:

- A low PFD frequency is acceptable, since this may simplify the FREF routing on the HTOL board.

Test Operation:

- VDDREF and VDDPOST at +20%.
- VDDHV at +10%.
- Minimum of 168 hours (1000 hours is common).
- Monitor VDDHV current and LOCK signal frequently, if possible.
- If LOCK=0 or VDDHV current increases by more than a few mA, all PLLs should be reset before continuing the test.

Some PLLs, especially under HTOL conditions, may be susceptible to an overdrive state in which the feedback divider misses edges, causing the VCO to speed up until it rails. If the PLL goes into this overdrive state for too long, it may be permanently damaged and the HTOL test may fail. To avoid this scenario, it is recommended to set PD = 1'b1 and then 1'b0 (or PLLEN = 1'b0 and then 1'b1) every 1 ms or less. This potential for PLLs with IO voltage supplying the VCO to be programmed into the "overdrive state" exists whether the chip is being tested for HTOL or in normal operation and must be avoided.