Programming Options for Si5332

The Si5332 is a part with three multisynth (fractional dividers) as seen in the block diagram below. This application note outlines the ways in which these dividers can be programmed.

KEY FEATURES OR KEY POINTS

- NVM programming
- Volatile memory programming
- Frequency-on-the-fly programming for ADPLL applications

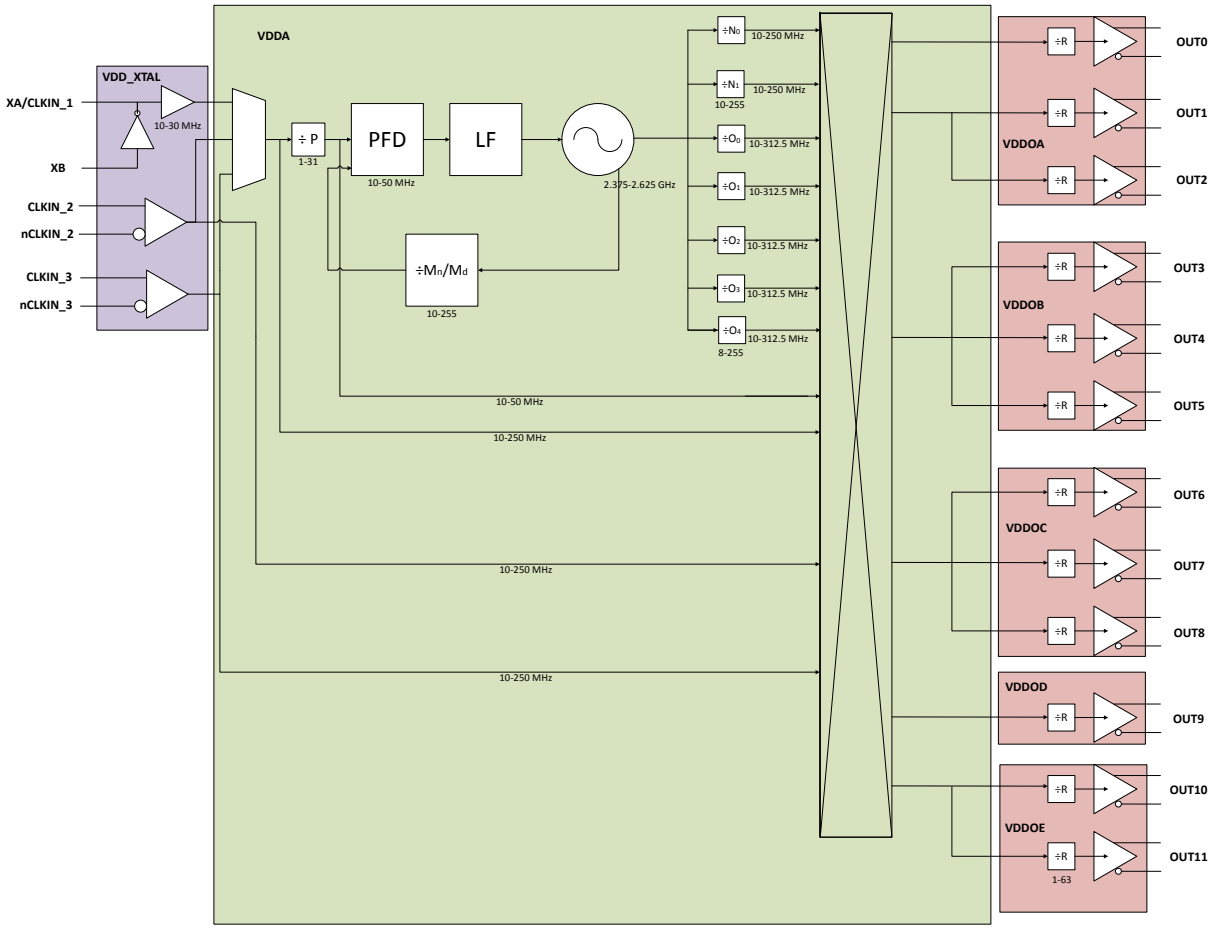


Figure .1. Si5332 PLL Block Diagram

1. Overview

Broadly speaking, there are three different methods to program the Si5332:

1. Frequency plan Non-volatile Memory (NVM) program: A configuration (i.e., a set of output frequencies) are required on power up.
2. Frequency plan volatile memory program: Si5332 powers up with a configuration A that needs to be modified to another configuration B.
3. Frequency on the fly program: Si5332 powers up with a configuration A and needs only a selected output (or outputs) to change while the other outputs retain their frequency and function.

Each method has its use case and all of them sometimes mean “re-programming” the Si5332 PLL. Hence, it is useful for the user to review the differences between each method and choose the initial configuration. As will be shown, the choice of the initial configuration determines the feasibility of methods 2 and 3 in the above list.

2. Frequency Plan NVM Program

The Si5332 Non-Volatile-Memory (NVM) is a sophisticated architecture. There are about 600 bytes of NVM memory available for about 250 bytes of register fields (the register fields that define a user's configuration completely). Hence, a user has the following options:

1. Configure an initial plan and order an NVM-programmed part from Silicon Labs.
2. "Re-configure" the NVM as many times as possible (the accounting of NVM already used and now available for NVM reprogram is managed by CBPro when the device is connected to CBPro through the CBPro dongle).

A register that is already programmed using an initial NVM can be selectively changed without affecting any other register and thereby saves memory space and re-programming capability for future needs. This memory management (available in Si5332) allows flexibility to users. The flow chart in the figure below illustrates the options available for NVM programming.

The following characteristics need to noted when changing NVM programming:

1. An NVM program (or re-program) takes effect only after a power cycle.
2. The power cycle needed means that "all output" clocks will be temporarily out due to the power cycle.
3. Hence, using NVM re-program when a dynamic frequency change is needed is not advisable

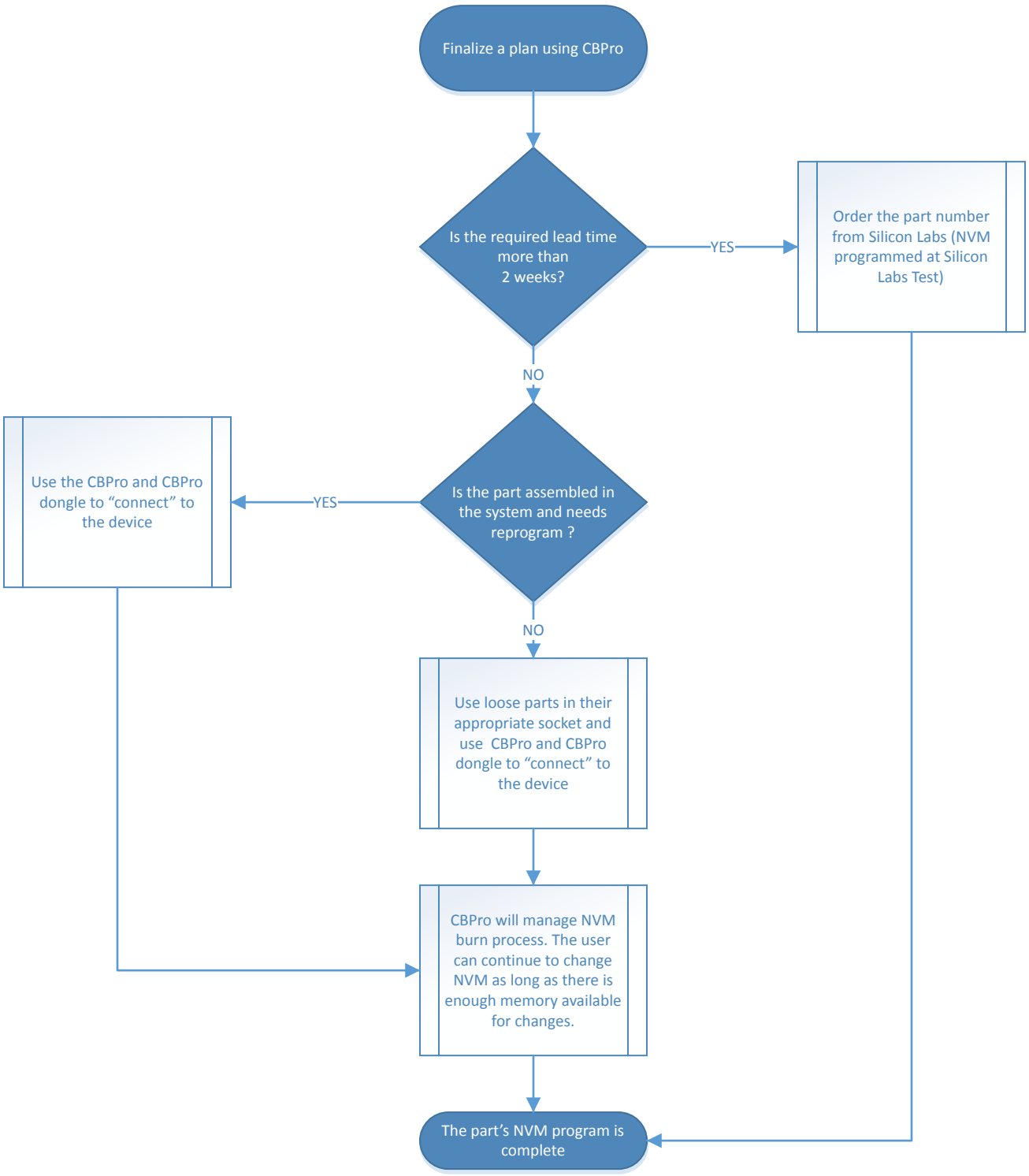


Figure 2.1. Si5332 NVM Programming

3. Frequency Plan Volatile Memory Program

The Si5332 registers needed to implement a configuration change must be determined by the user. There are two methods available to the user:

1. Use CBPro to create a new project file, export the register map, and list a difference (using CBPro diff tools for multiple projects or otherwise) in the registers that need reprogramming for the new configuration.
2. Use methods similar to the suggested method in the Si5332 Family Reference Manual to calculate all the register fields needed to implement the configuration.

When a comprehensive list of registers is generated, the user must identify the mode of the device that needs to be used for the re-program.

The Si5332 has two modes of operation:

1. READY mode: The mode in which the Si5332 digital system is ready to implement changes to any register. However, in this mode, output clocks are disabled.
2. ACTIVE mode: The mode in which only a select few registers can be programmed (as listed in the register map in the Si5332 Family Reference Manual). The outputs remain enabled in this mode but only a few registers can be reprogrammed.

In ACTIVE mode: The following register writes are needed for re-programming:

1. Write all the relevant registers as calculated from the steps above.

In READY mode: The following pre-amble and post-amble register writes are needed for re-programming:

1. Write 0x01h to register 0x06h and put the Si5332 into the READY state.
2. Write all the relevant registers as calculated from the steps above.
3. Ensure that the valid input clocks are available for the Si5332 to attempt a PLL lock.
4. Write 0x02h to register 0x06h and put the Si5332 into the ACTIVE state.

The volatile memory program (or re-program) is suitable for the following user applications:

1. Complete volatile memory program control: When the system designer needs complete independence in setting the output clock frequencies and/or output formats to suit varying system needs.
2. Minor modifications for system diagnostics: When the system designer needs to make minor modifications such as enabling or disabling an output or control presence of absence of spread spectrum clock, etc.

Note: Volatile memory program cannot guarantee presence of output clocks while register programming is underway.

4. Frequency-on-the-fly Updates to Output Frequencies

A frequency-on-the-fly (FOTF) update for an output frequency is depicted in the figure below. The characteristics of FOTF updates are:

- The frequency change is gltichless as shown in the figure below.
- When a certain output is being updated, other outputs remain unchanged.

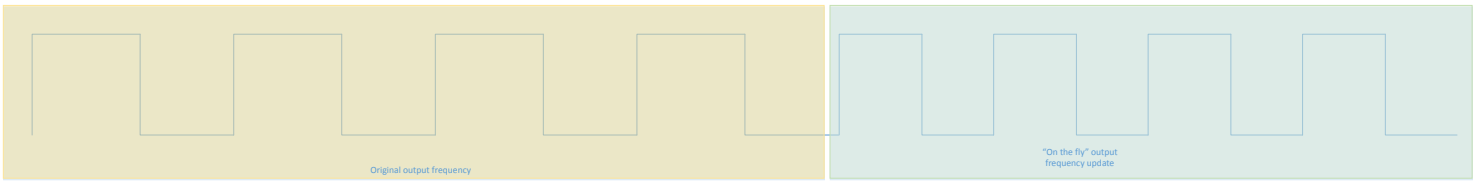


Figure 4.1. Illustration of an FOTF Update

Figure 4.2 Output Crossbar Associating Output Dividers with Outputs on page 7 below shows the “output” crossbar that connects the output dividers to various outputs. As described in the Si5332 Family Reference Manual, any output is derived as the ratio of the VCO frequency and the product of the output divider and R-divider associated with that output.

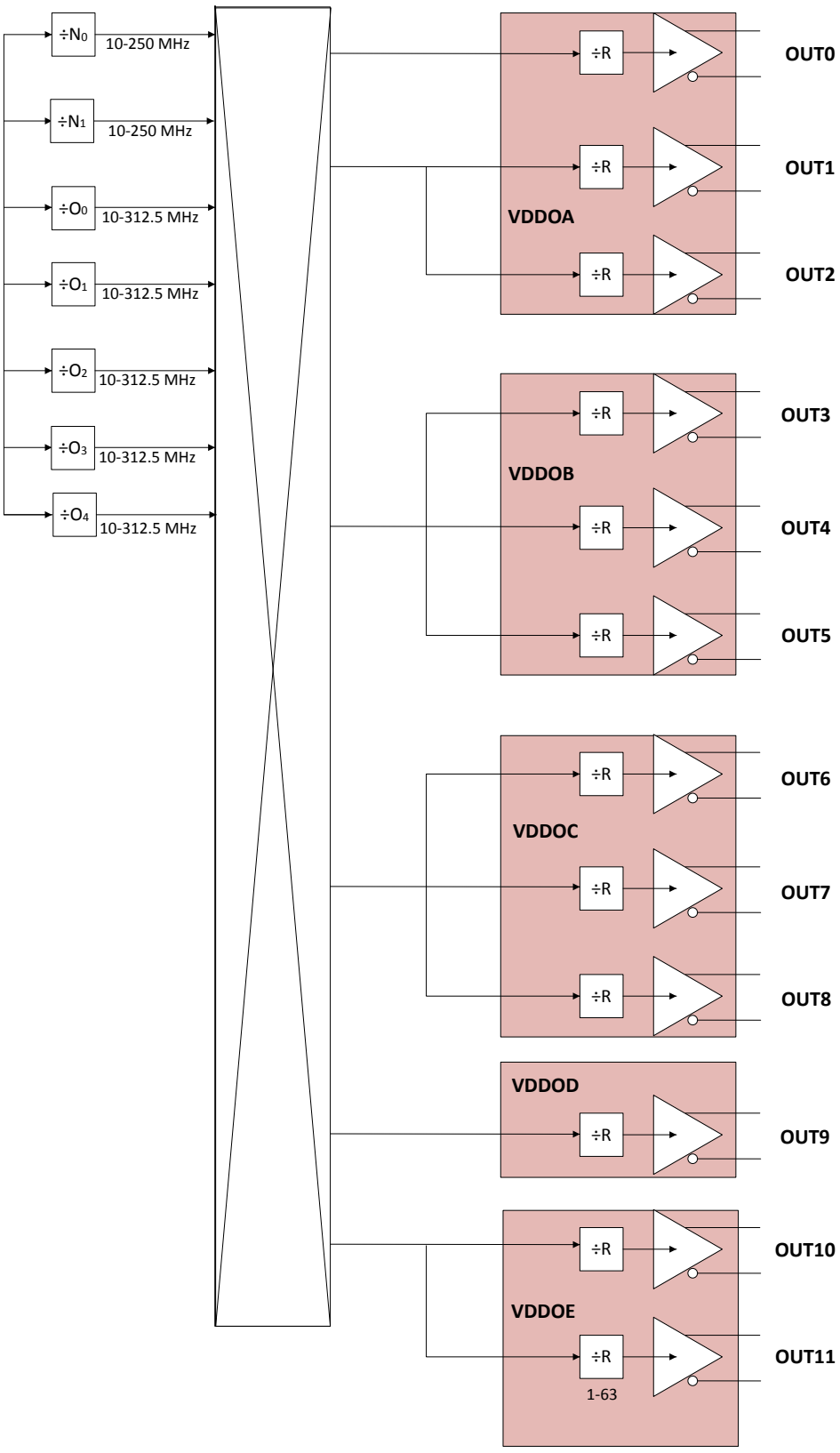


Figure 4.2. Output Crossbar Associating Output Dividers with Outputs

The dividers Nx (fractional multisynth dividers) and Oy (high speed integer dividers) have two banks, “bank a” and “bank b” with each bank holding an independent divider value. Only one bank (typically “bank a”) is the active divider bank, i.e., the bank that is currently driving the output. The R-dividers have only value and each output is driven by a dedicated R-divider. For FOTF applications:

1. The R-divider cannot be updated.
2. The “bank b” divider value can be updated and then a register updated to the IDx_DIV_SEL or the HSDIVy_DIV_SEL bit to 1. (If “bank b” is the active divider, update “bank a” and set the appropriate DIV_SEL bit to 0).

The section above outlines “how” to implement a frequency-on-the-fly update. The following section describes the two types of applications that need FOTF updates and outlines recommendations for optimal implementations. The two main applications that need FOTF are:

1. Frequency margining applications and/or coarse frequency tuning: In digital systems, the reference clock is sometimes increased (or decreased) in the order of 5-10% to test the maximum speed of the logic in the digital system. In display applications, the pixel frequency is increased in the order of a few MHz to accommodate better screen resolutions. Such applications need a “coarse” update to the frequency of the outputs. The high-speed integer dividers (Oy dividers) can be used for such applications. Although these dividers can only be updated in integer steps in the range of 10-255, they will meet the requirements of coarse frequency updates.
2. Synchronization and/or fine frequency tuning: In synchronization applications, where the local reference clock needs to be aligned to a master (such as MPEG encoding for video, synchronous Ethernet standards, etc.), a fine frequency step is needed (typically in steps of 1 ppm). The Nx dividers will provide a resolution as low as 10 ppb for such applications and are suitable for such systems. An example of the synchronization systems (usually implemented as a servo PLL in systems) is shown in Figure 4.3 Servo Loop to Implement Synchronization on page 8:

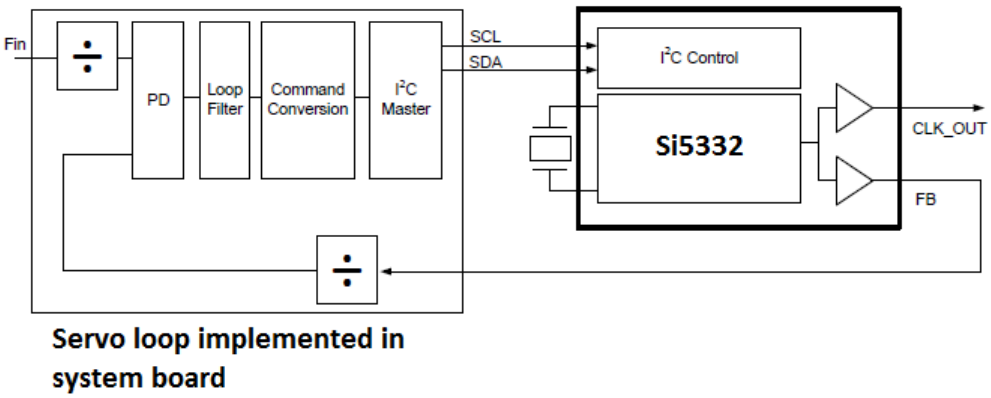


Figure 4.3. Servo Loop to Implement Synchronization

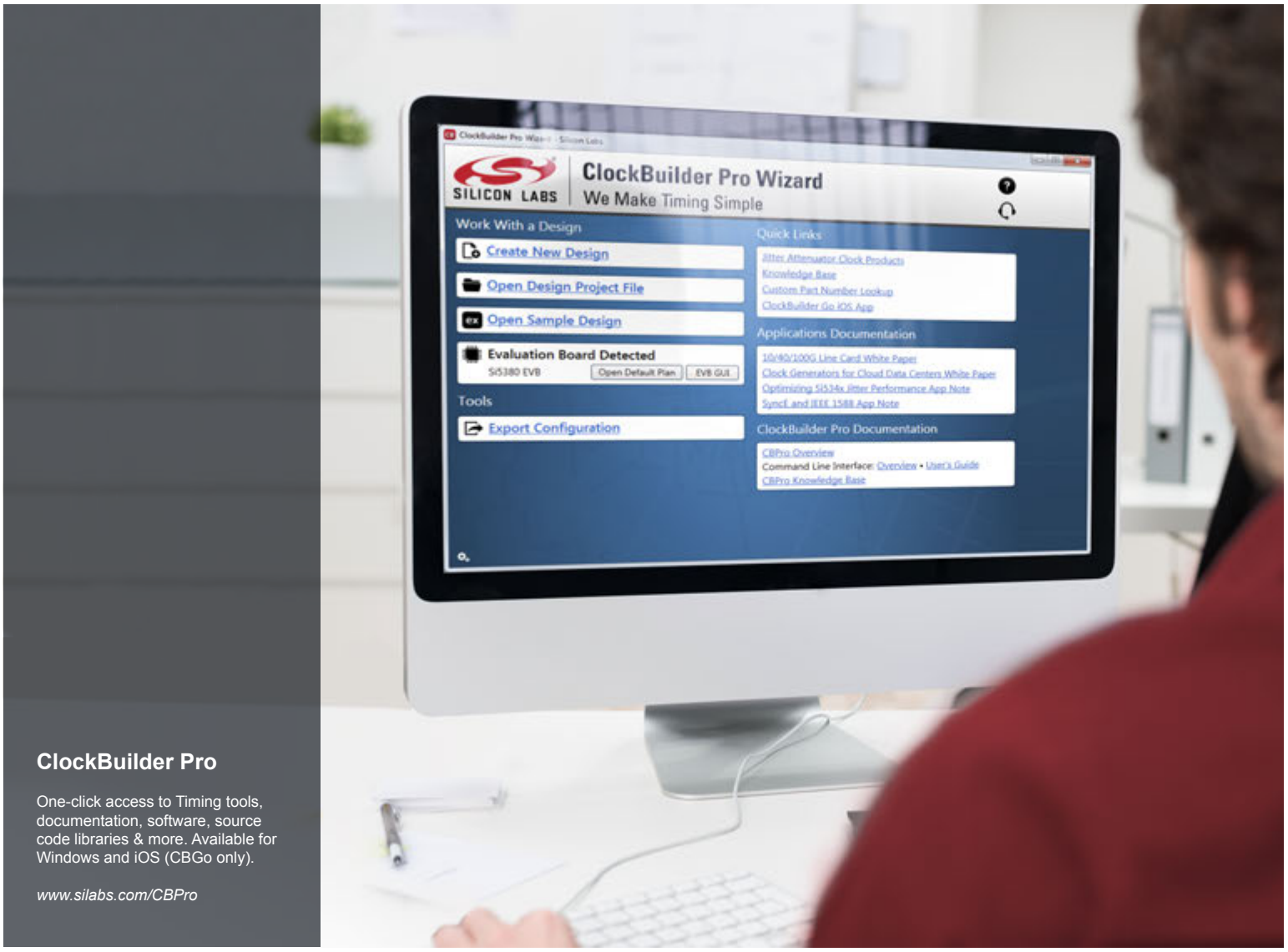
The synchronization achieved by servo loops is usually implemented as All-digital-PLLs (ADPLLs) in systems. When designing ADPLLs, the Si5332 is used as a digitally controlled oscillator (DCO). However, an I^2C update to Si5332 needs to be modelled as a delay. There is a delay of ~150 µs for an I^2C update for the remainder bytes and the IDx_DIV_SEL register field. If the step size is 10 ppb and the error that needs compensation is ~1 ppm, then the delay increases to about 15 ms. Hence, it is important that the step size and the error size are similar. The Si5332 offers the advantage of being able to correct for the exact error in a single I^2C write sequence. Hence, it is possible for users to allow a 150 µs delay in the system. Another advantage in using the Si5332 (or any multisynth based clock generator) is that the range of the DCO and associated “KVCO” for the PLL system is flexible and can be set by the user according to system needs. This is not available in traditional VCSO/VCXO units. Given the need to account for the delay of 150 µs, the largest BW for the servo PLL cannot be more than 650 Hz. Typically, servo PLLs have bandwidths in the order of 10-100 Hz and therefore, this constraint is not limiting for such designs.

5. Conclusion

The Si5332 is a versatile clock generator that can support various levels of programming including frequency on the fly applications. It can be used for frequency margining for both coarse and fine frequency adjustments and also be re-programmed (NVM and/or volatile memory re-program) to match system design changes. The table shows the cases when each of the three different programming options can apply. A single IC can be used for any (or all) of these needs in a system and can be used to reduce system board BOM, power, and area with increased functionality. The Si5332 can be used not just as a clock tree in chip solution, but also as the most flexible clock solution in the system.

Table 5.1. Summary of Programming Methods vs System Needs

System Need	VM	NVM	FOTF
System board re-designs/upgrades	Suitable	Suitable	Not suitable
System board modifications	Not suitable	Suitable	Not suitable
System clock speed test	Suitable	Not suitable	Suitable
Servo PLLs	Not suitable	Not suitable	Suitable



ClockBuilder Pro

One-click access to Timing tools, documentation, software, source code libraries & more. Available for Windows and iOS (CBGo only).

www.silabs.com/CBPro

Timing Portfolio
www.silabs.com/timing

SW/HW
www.silabs.com/CBPro

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Disclaimer
Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

Trademark Information
Silicon Laboratories Inc.® , Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOmodem®, Micrium, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress®, Zentri and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701