AN959: DCO Applications with the Si5341/40

Generically speaking, a DCO is the same thing as a numerically controlled oscillator (NCO) or a direct digital synthesizer (DDS). All of these devices are oscillators in which the output frequency is controlled by a digital input, typically a multi-byte word.

KEY POINTS

- Multiple independent DCO's can be implemented
- FINC/FDEC and Direct Write DCO modes available
- ClockBuilder Pro supports both DCO modes

1. Introduction

The Si5341/40 devices can be used as a multi-channel Digitally Controlled Oscillators (DCO's) for applications such as:

1. FIFO management
2. Frequency margining
3. Variable Frequency Local Oscillator

The DCO function is implemented by changing individual output fractional N dividers (See figure below) in one of two ways:

1. By directly writing to the N divider registers via the serial port
2. By using Frequency Increment (FINC) and Frequency Decrement (FDEC) pins or register bits to change the Nx divider by an amount specifed by the internal Nx_FSTEPW parameter.

Changing the divide value of one of the N dividers will have no effect upon the other N dividers. Hence the Si5341 allows five independent DCO's in a single package, the Si5344 allows four independent DCO's in the same package and the Si5340 allows four independent DCO's in the same package. The valid range of the N dividers is 10 to 4095 and the N divider value = N_NUM / N_DEN where N_NUM is a 44 bit value and N_DEN is a 32 bit value. Because the N_DEN term is 32 bits and the N_NUM term is 44 bits, changing the N_DEN term has much less resolution than changing the N_NUM term by approximately a factor of 2^{12} = 4096. When the Nx_NUM term is changed by any amount, the output frequency change will be glitchless. If the N_DEN term is changed there can be up to a 500 fs phase step; however, this can be avoided if the Most Significant bit (MSb) of the N_DEN term is not changed. Changing the N_NUM term causes an extremely small non-linearity that will not be a problem except for very rare applications. Changing the N_DEN term does not cause any non-linearity. This appnote will only describe changing the N_NUM term but if there is a specific need to change the N_DEN term then please contact Silicon Labs customer support at https://www.silabs.com/support/pages/contacttechnicalsupport.aspx. The DCO operation of the Si5345/44/42 device functions the same whether the device is in Holdover or Free Run or the normal locked state.

An N divider that is having its value written for DCO operation must be in the fractional mode of operation. If the DCO mode is being configured on a device in a system and the relevant N divider powers up in the integer mode, it will be necessary to put the N divider into the fractional mode and assert SOFT_RST. The setting name, PIBYP[4:0], controls the integer or fractional mode, PIBYP[N4 N3 N2 N1 N0]. When the respective bit of PIBYP is set to 0, the N divider is in fractional mode. For example, if all N dividers are to be used as DCOs, then set PIBYP = 0x00. A soft reset is required after changing any bit of PIBYP to a 0, and a soft reset will cause all output clocks to glitch in frequency and pulse off for a time. CBPro sets PIBYP as needed for all N dividers that have DCO enabled. Asserting SOFT_RST will cause all output clocks to turn off until the reset is finished. If ClockBuilderPro (CBPro) 2.1 or later is used to fully configure the DCO mode and create an Orderable Part Number (OPN) then the device will power up fully configured in the DCO mode and there will be no glitching of the output clocks. It is highly recommended that CBPro 2.1 or later is used when DCO mode is desired. The rest of this document explains the details of the algorithms used by CBPro to enable DCO operation and gives sufficient information so that DCO mode can be configured after a device powers up.

In CBPro you will not be able to enable a particular N divider for DCO operation (DCO page) unless one or more output frequencies are assigned to that particular N divider on the Define Output Frequencies page. Choosing an N divider for DCO usage on the Define Output Frequencies page also allows one or more outputs clocks to be assigned to a particular N divider that will be used as a DCO. In this manner multiple output clocks can be DCO controlled by the same N divider as long as the multiple output clocks have an even common multiple within the valid range of the N divider output frequency. An N divider that is assigned to be a DCO on the Define Output Frequency page will not be connected to any other output frequency.

CBPro should be used to configure the DCO mode because it will use advanced algorithms to minimize the output jitter across the entire DCO range. If CBPro is not used, the output jitter could be greater than 400 fs rms for some output frequencies within the DCO range.

DCO ranges of greater than 350 ppm should be avoided as higher jitter can occur in some cases. If you need a higher DCO range, please contact Silicon Labs.

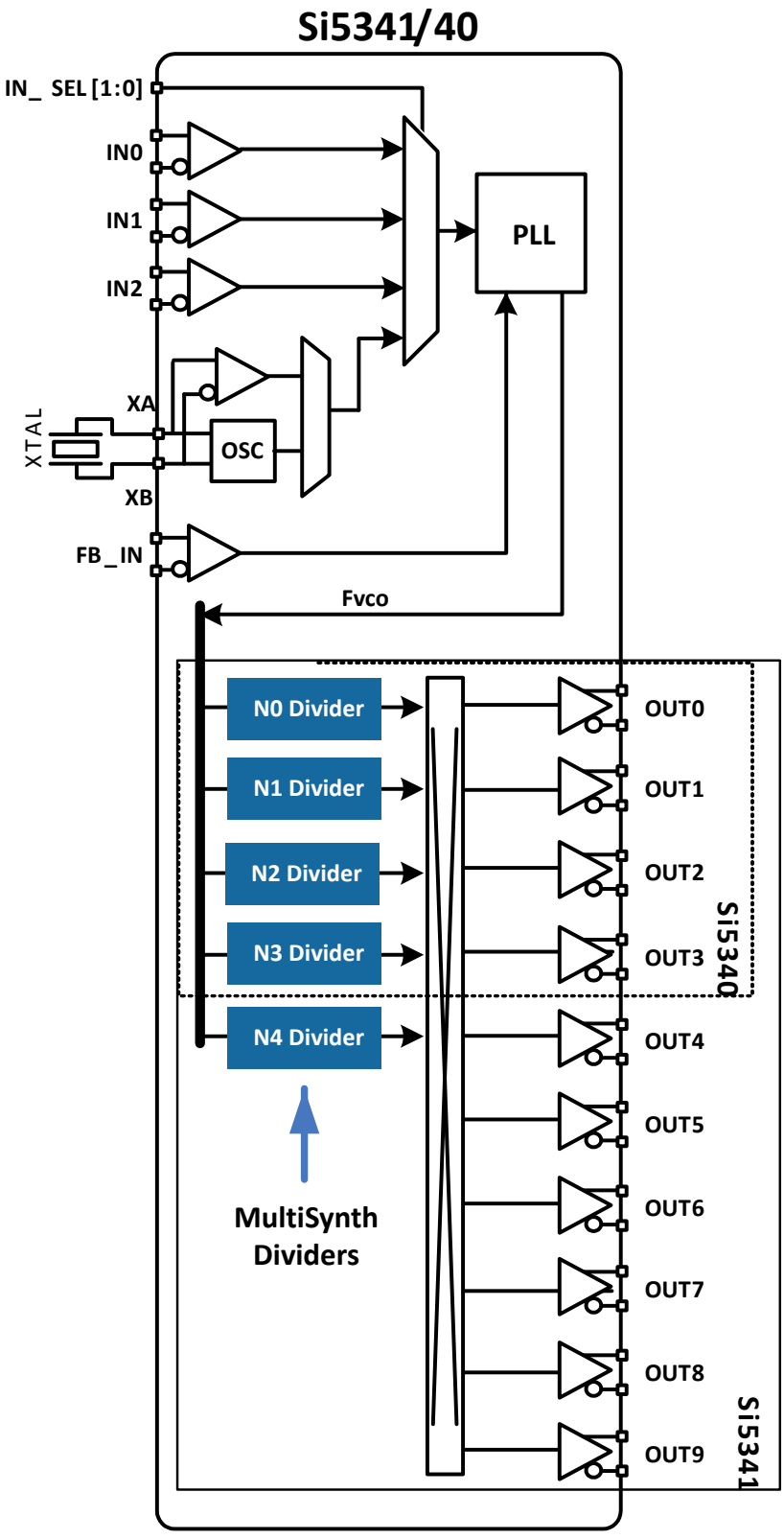

Figure 1.1. Functional Block Diagram

2. Equations

The following parameters are from the registers of the Si5341/40:

- Nx_DEN: Denominator of Nx divider, x = 0,1,2,3,4
- Nx_NUM: Numerator of Nx divider, x = 0,1,2,3,4
- Ry_REG: Register that determines the value of the Ry divider

2.1 Output Frequency Equations

$$Fout = \frac{Fvco}{Nx \times R}$$

Equation 1

where x is the index 0, 1, 2, 3, or 4 of the N divider, Fout is the output clock frequency in Hz and Fvco is the frequency of the VCO in Hz.

or

$$Fout = \frac{Nx\_DEN \times Fvco}{(Nx\_NUM \times R)}$$

Equation 2

where R is the output R divider value between the N divider and Fout (output frequency).

Note that increasing Nx_NUM will decrease Fout, and decreasing Nx_NUM will increase Fout. In addition, since Nx_NUM is in the denominator of equation 2, changes in Nx_NUM will cause a very slight non-linearity in the change of Fout. However since typical changes to Nx_NUM are miniscule compared the the intial value of Nx_NUM, the non-linearity is extremely small and of no consequence in all but very special applications.

An example of the % error in the step size of Nx_NUM follows.

For Fvco = 13500 MHz, R =2, Fout = 156.25 MHz, with N_NUM fully left shifted, the error in the step size is less than .001% for up to 1 million steps of the LSBit of the N_NUM. For this case, 1 million steps of the LSBit of the N_NUM is about 10,000 ppm. If this extremely small non-linearity is an issue for an application then contact Silicon Labs customer support at https://www.silabs.com/support/pages/contacttechnicalsupport.aspx to determine if changing the N_DEN term will work for your application.

2.2 Nx_NUM_delta Equations

For a desired output frequency step size, the Nx_NUM value will need to be added to or subtracted from by a value called Nx_NUM_delta. Nx_NUM_delta is calculated to be the value that will produce an output frequency step size that is as close as possible to the desired step size. The equation to calculate Nx_NUM_delta is slightly different depending upon whether the desired step is specified in Hz, ppm or ppb.

Note: If Nx_NUM_delta is calculated for Fout_step_Hz, then this only applies for the Fout that is in the equation. For example, if a 1000 Hz step is implemented on a 200 MHz output but a 100 Mhz output comes from the same N divider, the 100 Mhz output will step by 500 Hz.

Let Fout_step_Hz be the desired step size in Hz and Fout be in Hz, then the equation to calculate the Nx_NUM_delta is:

$$Nx\_NUM\_delta = round\left(\frac{Fout\_step\_Hz \times Nx\_NUM}{(Fout + Fout\_step\_Hz)}\right)$$

Equation 3

Let Fout_step_ppm be the desired step size in ppm, then the equation to calculate the Nx_NUM_delta is:

$$Nx\_NUM\_delta = round\left(\frac{Fout\_step\_ppm \times Nx\_NUM}{1e6 + Fout\_step\_ppm}\right)$$

Equation 4

Let Fout_step_ppb be the desired step size in ppb, then the equation to calculate the Nx_NUM_delta is:

$$Nx\_NUM\_delta = round\left(\frac{Fout\_step\_ppb \times Nx\_NUM}{1e9 + Fout\_step\_ppb}\right)$$

Equation 5

Note: ClockBuilder Pro versions 2.1 and 2.2 only support a step size in ppm, ppb or %.

2.3 Actual Step Size Equations

From Nx_NUM_delta, the actual initial step size (to increase the output frequency) in Hz is calculated as follows:

$$Actual step size in Hz = Fout \times \left(\frac{Nx\_num}{(Nx\_NUM - Nx\_NUM\_delta)} - 1\right)$$

Equation 6

Where Fout is in Hz.

From Nx_NUM_delta the actual initial step size (to increase the output frequency) in ppm is calculated as follows:

$$Actual step size in ppm = 1e6 \times \left(\frac{Nx\_NUM}{(Nx\_NUM - Nx\_NUM\_delta)} - 1\right)$$

Equation 7

From Nx_NUM_delta the actual initial step size (to increase the output frequency) in ppb is calculated as follows:

$$Actual step size in ppb = 1e9 \times \left(\left(\frac{Nx\_NUM}{(Nx\_NUM - Nx\_NUM\_delta)}\right) - 1\right)$$

Equation 8

3. Procedure for Using the Si5341/40 as a DCO with Direct Writes

CBPr0 2.1 and later will perform the calcutions for DCO with Direct Writes, however, what follows is the procedure to manually implement this same capability and give an understanding of what CBPro is doing.

3.1 Basic Procedure for Using the Si5341/40 as a DCO

1. Create a starting frequency plan using ClockBuilder Pro. Note the N dividers that are connected to the output clocks that you want to control as a DCO. When an individual N divider value is changed, all output clocks connected to that N divider will change by the same percentage amount.
2. Make sure that the N dividers that you are going to change have their fractional mode enabled.
    a. The setting name, PIBYP[4:0], controls the integer or fractional mode, PIBYP[N4 N3 N2 N1 N0]. When the respective bit of PIBYP is set to 0 the N divider is in fractional mode. If all N dividers are to be used as DCO's then set PIBYP = 0x00. A soft reset is required after changing any bit of PIBYP to a 0. Note that a soft reset will cause all outputs to glitch. CBPro sets PIBYP as needed for all N dividers that have DCO enabled.
3. Compute the Nx_NUM_delta for your required frequency step size.
4. Write the new Nx_NUM word. New Nx_NUM = old Nx_NUM +- Nx_NUM_delta.
    a. Add Nx_NUM_delta to Nx_NUM to decrease the output frequency and subtract Nx_NUM_delta from Nx_NUM to increase the output frequency.
    b. Write the new Nx_NUM value.
    c. Write the Nx_UPDATE bit to cause the change to take effect.

3.2 Detailed Procedure for Using the Si5341/40 as a DCO

1. Create a starting frequency plan using ClockBuilder Pro.
   a. On the Define Output Frequencies page of CBPro, make sure that you manually assign an N divider to each output frequency that you want to control as a DCO. It does not matter which N divider is chosen. Two or more output frequencies that have an even common multiple in the range of 200 Hz to 1425 MHz can be assigned to the same N divider and then all of these outputs will change when the N divider is written.
   b. Before the DCO option is enabled in CBPro, it will look for the lowest jitter combination of VCO frequencies and N divider values. When the DCO option is enabled in CBPro 2.1. and later, it will take into account the DCO range and again look for the best combinations of VCO frequencies and N divider values. In some cases, CBPro may not be able to find the most optimum frequency plan for jitter and will in this case warn that the output jitter may be higher although the maximum output jitter should always be less than 400 fs rms. If this happens, contact customer support at https://www.silabs.com/support/pages/contact-technicalsupport.aspx as a better solution may be possible.
   c. If using CBPro version 2.1 or later you can skip to step 3 below.
   d. The Nx_NUM and Nx_DEN values will be maximally left-shifted by CBPro.
   e. Write down the values of Nx_NUM for the N dividers that will be used as a DCO.
2. Compute the Nx_NUM_delta for your required frequency step size.
   a. Nx_NUM_delta is the value that must be added to or subtracted from the Nx_NUM value to get the desired frequency step size.
   b. Use equation 3, 4 or 5 to compute Nx_NUM_delta. CBPro version 2.1 or later will compute the Nx_NUM_delta value and show this in the DCO Details report.
3. Write the new Nx_NUM word. New Nx_NUM = old Nx_NUM +- Nx_NUM_delta.
   a. Do not change the Nx_DEN.
   b. Add Nx_NUM_delta to Nx_NUM to decrease the output frequency and subtract Nx_NUM_delta from Nx_NUM to increase the output frequency.
   c. All 6 bytes of the Nx_NUM may need to be written to account for byte boundaries being crossed as the Nx_NUM value is increased or decreased.
   d. Make sure that all N dividers that are being written have their fractional mode enabled.
      i. Read N_PIBYP[4:0] = 0x0A04[4:0] to see if the N divider that is being written to is in the fractional or integer mode. A 1 in this field bypasses the Phase Interpolator of the respective N divider and puts it into the the integer mode.
      ii. If needed, write the respective bit of N_PIBYP = 0 to change the necessary N dividers to the fractional mode.
      iii. Write SOFT_RST = 1 (0x001C).
   e. Write the new N_NUM value.
      i. For N0_NUM write to registers 0x0302 to 0x0307. Write N0_UPDATE bit 0x030C = 1 to cause the new N0_NUM value to take effect.
      ii. For N1_NUM write to registers 0x030D to 0x0312. Write N1_UPDATE bit 0x0317 = 1 to cause the new N1_NUM value to take effect.
      iii. For N2_NUM write to registers 0x0318 to 0x031D. Write N2_UPDATE bit 0x0322 = 1 to cause the new N2_NUM value to take effect.
      iv. For N3_NUM write to registers 0x0323 to 0x0328. Write N3_UPDATE bit 0x032D = 1 to cause the new N3_NUM value to take effect.
      v. For N4_NUM write to registers 0x032E to 0x0333. Write N4_UPDATE bit 0x0338[0] = 1 to cause the new N4_NUM value to take effect.
      vi. The value of Nx_NUM can be read back at any time.
      vii. A write to N_UPDATE_ALL 0x0338[1] = 1 will cause all N dividers to update.

3.3 Example of DCO with Direct Write

For the following input and output frequencies:

Fin = 19.44 MHz

Fout0 = 156.25 MHz

With the following DCO parameters:

Step size = 2.5 ppm

Range = +- 200 ppm

CBPro 2.1 computes the following:

Fvco = 13500 MHz

P0 = 18

R0 = 2

N0 = 43.2

N0_NUM = 0x1B00000000 = 115,964,116,992

N0_DEN = 0xA0000000 = 2,684,354,560

Only 1 N divider is being used.

N_PIBYP[4:0] = 00001; N0 is in integer mode.

Write N_PIBYP[4:0] = 00000b.

Write SOFT_RST = 1 (0x001C).

Using equation 4 with Fout_step_ppm = 2.5, we can calculate N0_NUM_delta as:

N0_NUM_delta = round (2.5 x 115,964,116,992/(1e6+2.5) = 289,910 (0x46C76).

The exact initial frequency step up in ppm with this N0_NUM_delta can be found from equation 7.

Exact frequency step in ppm = 1e6 x ((N0_NUM / (N0_NUM + 262,145)) - 1) = 2.500003727812 ppm.

To increase the original frequency by 2.5 ppm it is necessary to write the N0_NUM value to:

N0_NUM (new) = N0_NUM (original) - N0_NUM_delta = 115,963,827,082 (0x1AFFFB938A)

then write N0_UPDATE = 0x030C = 1 to cause the new N0_NUM divider value to take effect.

To help make calculations with rational fractions such as N_NUM / N_DEN, CBPro has a rational fraction calculator. This calculator is accessed from each of the configuration pages by clicking on the round gear icon just to the right of the version number at the top left of the page.

4. Using the Si5341/40 as a DCO with FINC and FDEC

The Si5341/40 have FINC and FDEC register bits that when written high cause a frequency increment or decrement respectively on specific output clocks. These register bits are self-clearing. The Si5341 also has FINC and FDEC pins that perform the same function when pulsed high (>100 ns pulse). When the FINC or FDEC pins are pulsed high (>100 ns pulse) the output frequency will increase or decrease respectively.

The amount of frequency increase or decrease is controlled by the Frequency Step Word (Nx_FSTEPW) for the selected Nx divider(s). Nx_FSTEPW is equivalent to the Nx_NUM_delta parameter used for Direct Writes except that Nx_FSTEPW is stored in the device registers and the addition/subtraction to the numerator of the Nx divider is implemented by the device. There is a separate Nx_FSTEPW for each Nx divider and all outputs derived from that Nx divider will be affected in the same proportion. Nx_FSTEPW will add to Nx_NUM when FDEC is asserted and will subtract from Nx_NUM when FINC is asserted. The Nx_FSTEPW can be calculated exactly as was done for the Nx_NUM_Delta above. The calculated Nx_FSTEPW must be written to the register so that the FINC and FDEC know what to add and subtract from the Nx_NUM value. Note that because of the design implementation, a read of Nx_NUM will remain the same regardless of how many times FINC or FDEC is pulsed. For this reason it may be important that the FW/SW keep track of the expected Nx_NUM value due to the total number of FINC and FDEC commands that have occurred. To restore the output frequency back to the value defined by the Nx_NUM and Nx_DEN register values, write Nx_UPDATE = 1.

FINC and FDEC can affect any number of the N dividers, there is a mask field N_STEP_MSK[4:0] that prevents FINC and FDEC from affecting N dividers that have their mask bit set. For example if FINC and FDEC are only to affect the N3 divider, then set N_STEP_MSK[4:0] = 10111b. If N_STEP_MSK[4:0] = 00000b, then an FINC or FDEC will cause all N dividers to be affected by their respective Nx_FSTEPW words. Note that if Nx_FSTEPW = 0 and this Nx divider is not masked, a FINC or FDEC assertion will not change or glitch the outputs that are connected to this Nx divider.

The registers described above are shown in the table below.

Table 4.1. Name Settings for FINC and FDEC Operation

Name	Register Address	Type
FINC	0x001D[0]	S*
FDEC	0x001D[1]	S*
N0_FSTEPW	[0x0340[43:40]…0x033B[7:0]	R/W
N1_FSTEPW	[0x0346[43:40]…0x0341[7:0]	R/W
N2_FSTEPW	[0x034C[43:40]…0x0347[7:0]	R/W
N3_FSTEPW	[0x0352[43:40]…0x034D[7:0]	R/W
N4_FSTEPW	[0x0358[43:40]…0x0353[7:0]	R/W
N_STEP_MSK	0x0339[4:0]	R/W
N0_UPDATE	0x030C[0]	S*
N1_UPDATE	0x0317[0]	S*
N2_UPDATE	0x0322[0]	S*
N3_UPDATE	0x032D[0]	S*
N4_UPDATE	0x0338[0]	S*

Note: *S means self-clearing.

CBPro 2.1 or later should be used whenever a DCO mode is required because:

- It will calculate the Nx_FSTEPW register word(s).
- It will optimize the jitter (if needed) for all outputs based upon the DCO range(s) that are input.

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

SILICON LABS®

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

http://www.silabs.com