User Guide for DAC39RF10 DUC Simulink Model



Figure : DUC Model with Stimulus and Analysis

Table of Contents

[1 Overview 2](#_Toc79402418)

[2 Terms and Definitions 2](#_Toc79402419)

[3 How to run the model 3](#_Toc79402420)

[3.1 Changing the Stimulus 4](#_Toc79402421)

[3.2 Changing DUC Parameters 5](#_Toc79402422)

[4 Features and Limitations 8](#_Toc79402423)

[5 Revision History 9](#_Toc79402424)

# Overview

This user guide explains how to use the DAC39RF10 DUC Simulink model. The model includes key features of the DAC39RF10 Digital Up-Converter such as:

* Up to four IQ input channels supported
* Each channel is interpolated by a factor of 2x to 256x
* Each channel includes an NCO and Mixer
* Supports real or complex mixer output (analogous to DUC\_FORMAT register)
* Supports adjustable DUC output gain (analogous to DUC\_GAIN register)
* Supports routing any DUC channel to either DAC output (analogous to DAC\_SRC register)

The model includes a test bench (harness) that demonstrates the model. This test bench includes a stimulus generator, the DUC itself, an output scope, and post-simulation FFT analysis.

# Terms and Definitions

| **Term** | **Definition** |
| --- | --- |
| DUC | Digital Up-Converter. Consists of interpolation and mixing. |

# How to run the model

1. Ensure that your system has MATLAB installed. The model was developed with MATLAB R2018a, and should work on newer versions of MATLAB.
2. Launch MATLAB. Then launch Simulink by typing **simulink** into the command window, or clicking the Simulink button in the toolbar.
3. Open the model file (dac39rf10\_duc.slx). You should see a model schematic that looks like **Figure 1.**
4. You can immediately run the model with the default settings by clicking the **run** button (see figure below).  
   
5. The model will run and plot time and frequency domain results. There are three different output plots:
   1. The ‘Output Scope’ will plot the time domain output of the model, including two channels. The two channels correspond to the two DAC channels of the DAC39RF10. See Figure 2.
   2. The ‘FilterOut’ scope will plot the interpolated input data (before mixing has occurred). This is provided for general debug purposes. See Figure 3.
   3. When the simulation completes, two MATLAB figures are created containing frequency domain plots of the output channels. See Figure 4.



Figure : Example Plot from 'Output Scope'



Figure : Example Plot from 'DUC/FilterOut' Scope



Figure : Example FFT Plots

## Changing the Stimulus

The user has two choices for the stimulus applied to each DUC input channel:

* Complex Exponential with specific amplitude and frequency
* User defined samples (i.e. “User Data”)

By default, a complex exponential is driven into each input channel. You can double-click on the ‘DUC Stimulus’ subsystem to modify the input stimulus. You can configure the input sequence length (N), as well as the amplitude and frequency of each input channel. A sequence length of N=512 is the default. Longer sequences will give more frequency resolution, but will take longer to simulate. When you modify the sequence length, the simulation stop time is adjusted automatically.

The input frequency of each channel can be adjusted from -0.5 to +0.5. The units are ‘cycles per sample.’ Values larger than 0.4 or less than -0.4 will be outside the passband of the DUC interpolators and may produce unexpected aliases. Therefore, it is recommended to keep the frequency between -0.4 and +0.4. The frequency you enter may also be rounded slightly to ensure the input completes a whole number of cycles in the defined sequence length.

To define user-supplied data, enter a MATLAB expression (or variable) into the **User Data** field for a channel. The expression is evaluated in the base workspace. The expression should evaluate to a row or column vector. Each element of the vector is a complex sample. With user-supplied data, the **Amplitude** and **Frequency** fields are ignored. To use them again, simply delete the expression in the **User Data** field.



Figure : Configuring Input Stimulus

## Changing DUC Parameters

The parameters of the DUC itself can be easily changed by double-clicking the DUC subsystem. You can configure a number of settings such as:

1. The interpolation factor. The supported values are: 2x, 3x, 4x, 6x, 8x, 12x, 16x, 24x, 32x, 48x, 64x, 96x, 128x, 192x, and 256x.
2. For the NCO/Mixer, you can configure the frequency and phase of each channel. You can also enable complex output (DUC\_FORMAT=1).
3. You can adjust the output gain of each DUC channel in the DUC\_GAIN tab. A gain of 0dB, -6dB, or -12dB is supported for each channel.
4. You can choose which DUC outputs are bound to each DAC channel. Use the DAC\_SRC0 tab to configure output channel 0. Use the DAC\_SRC1 tab to configure output channel 1.
   1. When DUC\_FORMAT=0, you can bind the real (in-phase) output of any DUC to a DAC output channel.
   2. When DUC\_FORMAT=1, you can bind the real (in-phase) or imaginary (quadrature) output of DUC0 or DUC1 to a DAC output channel. The output from DUC2 and DUC3 are not available. This scheme matches the scheme used in the DAC39RF10 DUC.
   3. When binding more than one DUC output to the same DAC, you should reduce the DUC gain to avoid saturation.



Figure : Configuring Interpolation Factor and NCO/Mixer Settings



Figure : Configuring DUC Output Gain



Figure : Configuring Output Binding (DAC\_SRC)

# Limitations

The model supports many of the features of the DAC39RF10 DUC, but has some limitations:

| **Item** | **Description** |
| --- | --- |
| Number of Input Channels | The model always supports 4 input channels, regardless of the interpolation factor. The real DAC39RF10 DUC is limited to 1 channel when interpolating by 2x or 3x, and is limited to just 2 input channels when interpolating by 4x or 6x. |
| Interpolation Factors | The model does not support 1x interpolation, but it does support all other interpolation factors that the DAC39RF10 DUC supports. |
| Sample resolution | The sample resolution of the interpolation filters and NCO/Mixer matches the DAC39RF10 DUC design, but due to some rounding differences in Simulink, this model is not bit-accurate to the DAC39RF10 DUC. |
| NCO Frequency | This model supports a 64-bit frequency. The user programs the frequency as a decimal number from 0.0 to 1.0. There is no need to multiply the number by 264. In most cases the user will want to choose NCO frequencies that produce bin-centered output frequencies and generate nice FFT plots. The programmed frequency should be in the form of FREQ= n/N\*L, where n is any integer, N is the input sequence length (default is 512), and L is the interpolation factor. Following this form will ensure the output completes a whole number of cycles in N\*L output sample periods, producing a nice FFT without spectral leakage. |
| NCO Hopping | The NCO in this model does not support any frequency hopping. The frequency remains constant during the simulation. |
| Latency | The latency of this model is not identical to the actual DAC39RF10 DUC. |
| DAC Mixing Modes | The DAC39RF10 supports various DAC outputs modes such as non-return-to-zero, return-to-zero, return-to-inverse, etc. This model does not include those modes. Its output is most similar to the NRZ or non-return-to-zero mode. |
| FFT Plots | When the model completes, FFT plots are generated of the output. The code that generates the plots can be viewed by clicking **File**->**Model Properties**->**Model Properties**. Then click the **Callbacks** tab and finally click **StopFcn** in the ‘Model callbacks’ pane to view the code. The code automatically determines the FFT length by first discarding the first 40\*L samples (where L is the interpolation factor). This accounts for the startup latency of the model. Then additional samples are discarded if necessary to arrive at L\*2M samples, where M is the largest integer possible, given the number of available samples. |

# Revision History

| **Version** | **Author** | **Date** | **Description** |
| --- | --- | --- | --- |
| 0.1 | Paul Kramer | 8/9/2021 | Initial version |