AD6676 Matlab Simulink Model User's Guide

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

Getting Started with the AD6676 Matlab Simulink Model

INTRODUCTION

The AD6676 Matlab Simulink model simulates the response of the AD6676's tunable bandpass ΣΔADC and digital signal processing block shown in figure 1 based on user-specified parameters shown in Table 1. Note that the functionality of the model has *only* been tested in the MATLAB & Simulink R2013a environment with a *protected model* created from this environment. The following R2013a licenses are required to run the model: MATLAB, Simulink, Control System Toolbox, DSP System Toolbox and Signal Processing Toolbox.

The model is a high-level functional representation of the AD6676 with the following device characteristics not modeled:

- ADC distortion performance with and without the shuffler enabled
- AGC loop and overload detection
- Anti-aliasing response
- Digital path round-off and overflow

The ADC's noise contribution has only been partially modeled but does not include any dependency of the IF center frequency (FO) or inductor value (L). The model processes the signal after it collects all the input sample points and processes them as an array of data. In other words, the model does not process the signal sample by sample. The 16-bit digital I/Q output is normalized to voltage having a max span of +/-1 V corresponding to 0 dBFS. Lastly, the AD6676's pipeline latency from IF input to digital output is captured in the model.

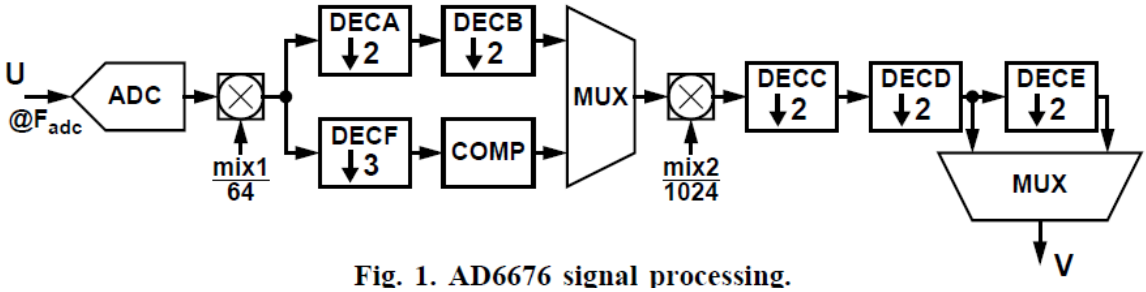


Fig. 1. AD6676 signal processing.

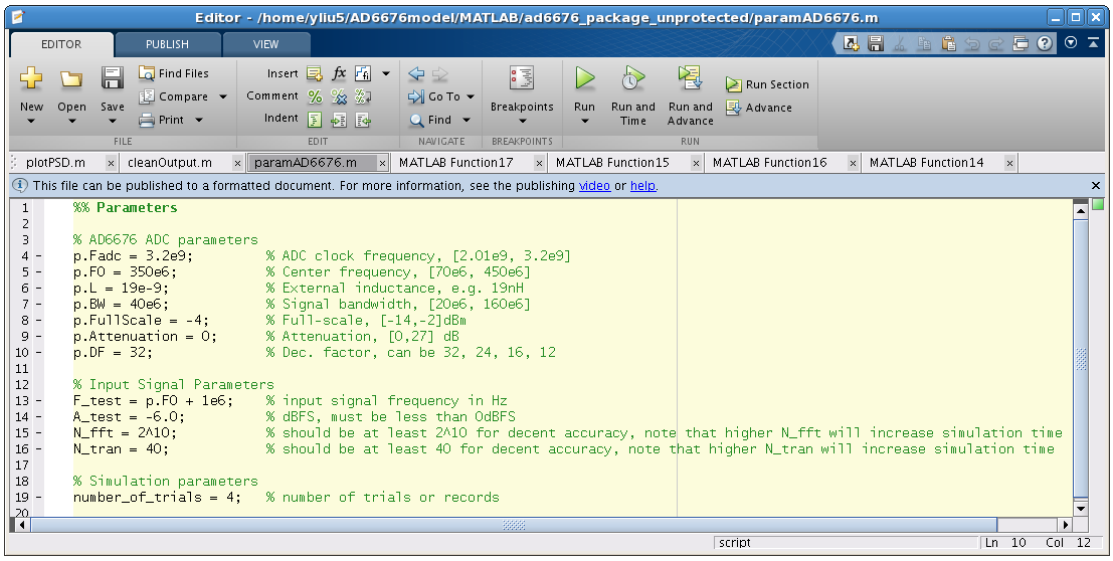
One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 www.analog.com
Fax: 781.461.3113

Name	Description
Fadc	ADC Clock Frequency
F0	IF Center Frequency
BW	IF Bandwidth
Attenuation	IF Attenuator Setting
FullScale	ADC Full-scale level
DF	Digital Filter Decimation Factor (12, 16, 24, 32)

Table 1. Selected Model Parameters

CONFIGURING MODEL AND INPUT PARMAETERS

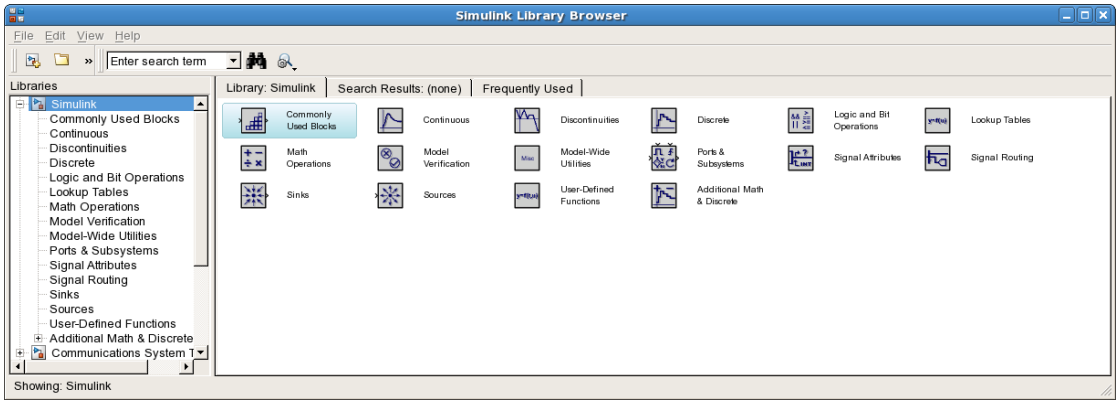
1. Open paramAD6676.m in a text editor



2. In paramAD6676.m under % AD6676 ADC Parameters, set the model parameters, which are described in *Table 1*. For example, to set the decimation factor to 32, set p.DF = 32;
   Note: please refer to the comments beside the parameters in paramAD6676.m for information regarding the allowed values of those parameters
3. In paramAD6676.m under % Input Signal Parameters, set variables A_test and F_test to configure the amplitude in dBFS and frequency in Hz of the input signal; for example, A_test = -6.0;
4. In paramAD6676.m under % Simulation Parameters, set the number of trials through the variable number_of_trials; for example, number_of_trials = 4;

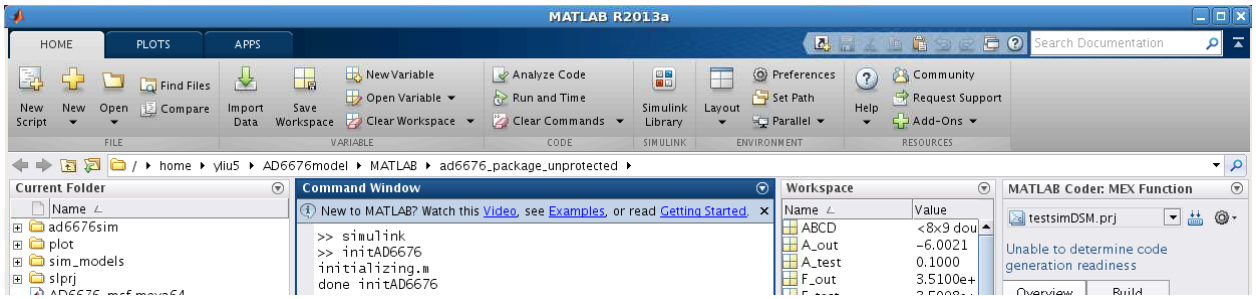
Running the Model

1. Open MATLAB
2. Go to ad6676_package_R2013a/
3. Type *Simulink* in the MATLAB command line to open up the Simulink Library Browser



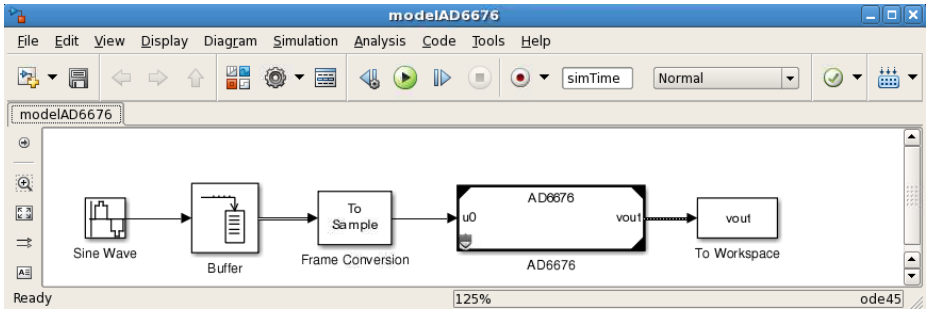
4. Type *initAD6676* in the MATLAB command line;

   note: initAD6676 processes the model parameters and sets paths so it should be typed again if you wish to run the model after the path settings or the model parameters have been changed



5. Open modelAD6676.slx through the Simulink Library Browser; if you wish to generate an input signal that is not sinusoidal, please replace the sine wave source block in modelAD6676.slx with a source block of your choice; you can also replace the ToWorkspace block with a sink block of your choice

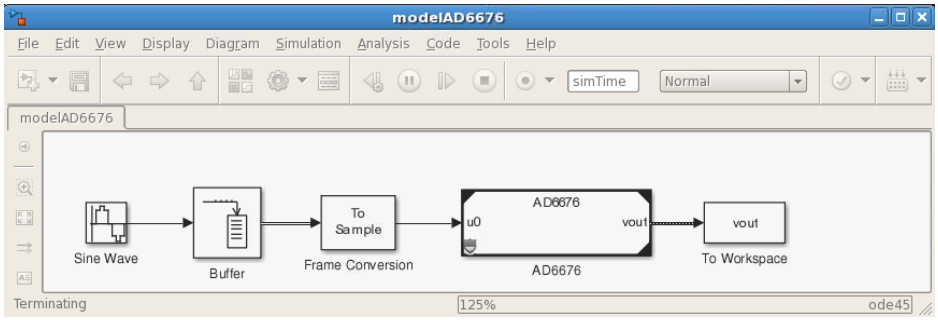
   note: if step 4 has not been completed yet, Simulink will say that the AD66676 block is not found



6. Click the green run button on the top of modelAD6676.slx; alternatively, you can type *sim('modelAD6676')* in the MATLAB command line.

   Note: please ignore the "### Model reference SIM target …" warning in the command line as long as

step 7 is being followed



7. To run the model again, type *initAD6676* in the MATLAB command line and then click the run button in Simulink;

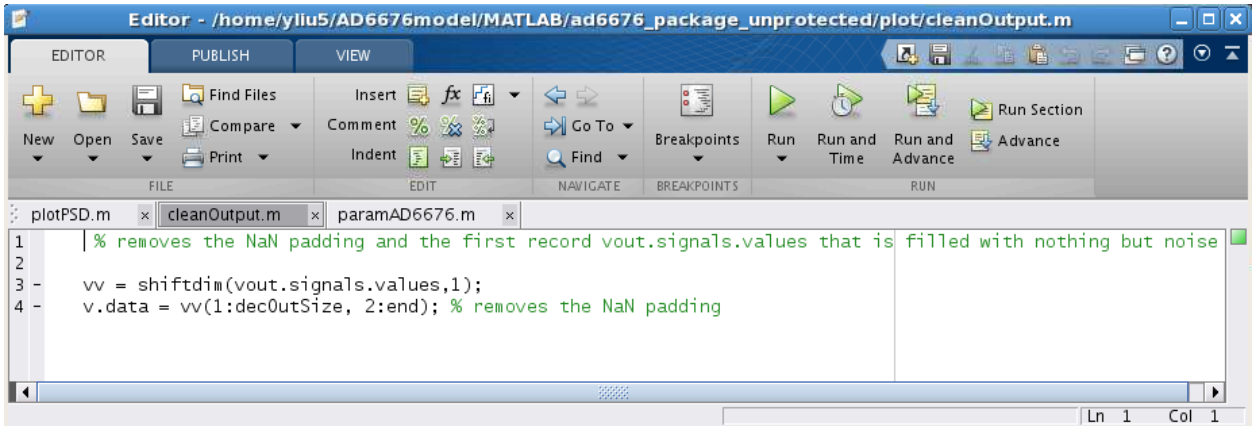
Warning: please do not press ctrl+c while the model is running; this may lead to unexpected behavior may occur and you may need to download the model again.

Analyzing the Output

How to analyze data:

1. To analyze the output, the model must first finish running, which happens when the run button of modelAD6676 turns back to green
2. The output data of the model is saved under the matrix vout.signals.values. Each column of this matrix is a trial of the AD6676 output.

   For example, ./plot/cleanOutput.m extracts all the valid output trials:



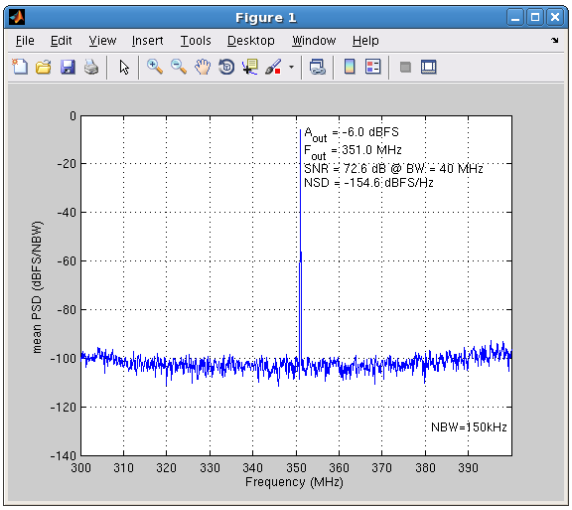
   Warning: 1:decOutSize is needed to remove the trailing NaN from Simulink. *decOutSize* is an integer variable that describes the length of the output signal.

   Warning: 2:end is needed because the first column is filled with only noise, since input samples have not reached the signal processing blocks by the time when the first column is generated

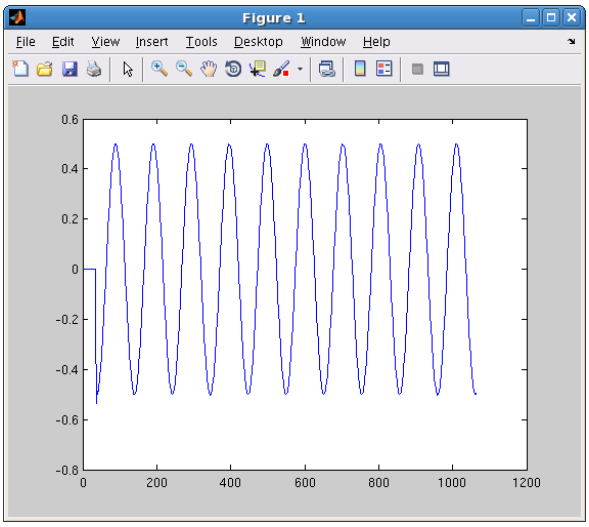
3. ./plot/plotPSD.m demonstrates an example of how vout can be analyzed. The script plots the average power spectral density of all output trials.

Some example plots of the output with a sinusoidal input at the frequency of F0+1MHz and DF=32, unless specified otherwise:

- plotPSD; % average power spectral density of all output trials



- plot(real(v.data(:,end))); % plots the real part of the last output trial

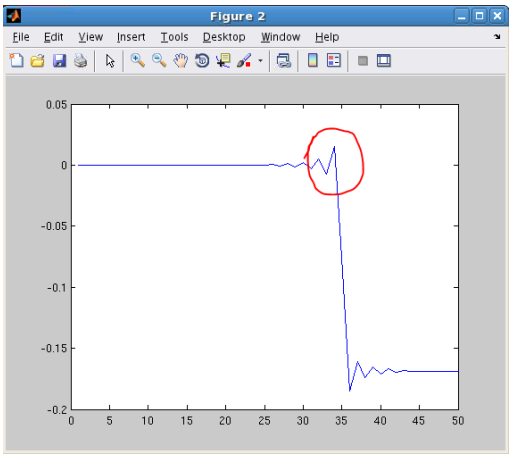


Note: the first few output samples are only made of noise due to the delay in the path. The total delay in samples along the data path is as follows:

DF	**32**	**24**	**16**	**12**
Delay in output samples	34	34	32	32

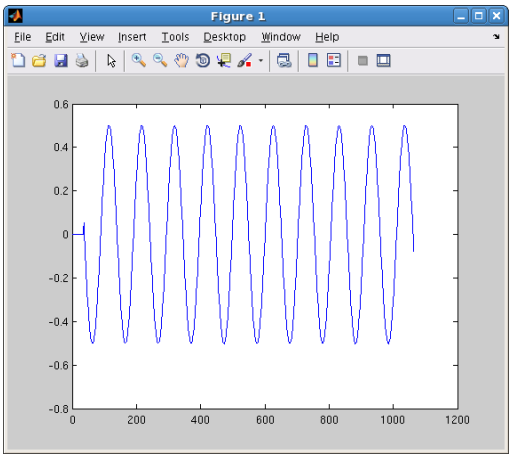
- plot(real(v.data(1:50,end))):

  The figure below demonstrates the pipeline delay of the AD6676 primarily resulting from the AD6676 digital signal processing path and (to a lesser extent) the JESD204B serializer. In this case, the frequency of the sine wave generator was set to the exact frequency of AD6676's digital NCO setting (F0) such that a delayed step response is evident in the envelope response seen at the IQ data output;

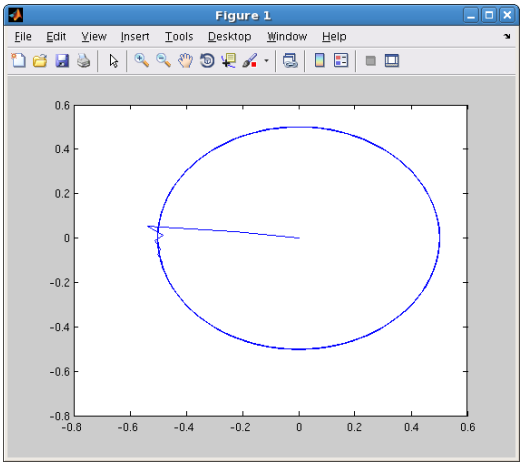


The red marker circles the 35th output sample, which is the first output sample that is not just made of noise, which is when the first input sample has reached the output. That demonstrates a latency of 34 output samples for the case of DF=32

- plot(imag(v.data(:,end))); % plots the imaginary part of the last output trial



- plot(v.data(:,end)); % plots the real part against the imaginary part of the last output trial



Note: those plots may be slightly different for each run even with the same model input and parameters because a random number generator is used in the noise model

5. Package Directories

Files in this directory:

- AD6676_msp.mexw64 is a file generated from compiling the model, as mentioned in *4. Analyzing the Model*
- initAD6676.p is a protected MATLAB file for initializing the model, as mentioned in *3. Running the model*
- modelAD6676.slx is a Simulink file used for running the Simulink model, as mentioned in *3. Running the Model*
- paramAD6676.m is a MATLAB file used by the user to set model parameters, as mentioned in *2. Configuring Model and Input Parameters*

Folders in this directory:

- ./ad6676sim/ contains all the protected MATLAB code used by this model
- ./sim_model/ contains all the protected Simulink models
- ./plot/ contains the scripts cleanOutput.m and plotPSD.m used for extracting the model output and analyzing its power spectral density, as mentioned in *4. Analyzing the Model*
- ./slprj/ contains files generated from compiling the model

6. Contact Support

If you have any questions regarding this Simulink model, please submit a question on EZONE (https://ez.analog.com).