Description, Installation, and Use of *vpulse*:
A Program to Create a Sampled Piecewise Linear Periodic Square Wave

Shawn M. Logan
March 18, 2025
smlogan@ieee.org

1. Background and Outline

In many signal processing and circuit simulation studies, periodic square waves serve as one or more input stimuli. For example, in the design and simulation efforts for an analog-to-digital converter (ADC), phase detector, or mixer, a periodic square wave may serve as the sampling clock, a reference clock input or a local oscillator respectively.

Simulations might be at the system level using a simulator such as Octave[1] or MATLAB®[2] or at the circuit level and use a SPICE based simulator. In either of these cases, an external file may be used to supply a time dependent stimulus.

In many cases, creating a set of sample points to form a piecewise linear description of a square wave does not pose a significant challenge. Creating more complex waves, such as amplitude or phase modulated square waves, can require far more effort and may be non-trivial.

The program described in this document, *vpulse*, is designed to create a set of samples representing a complex periodic square wave. In addition to allowing for programmable transient waveform parameters, the program includes the following features: the ability to amplitude or phase modulate a square wave; choose a deterministic or random noise modulation source; and include an optional first-order post filter to allow one to limit the square wave's bandwidth and remove the discontinuities inherent in a piecewise linear waveform.

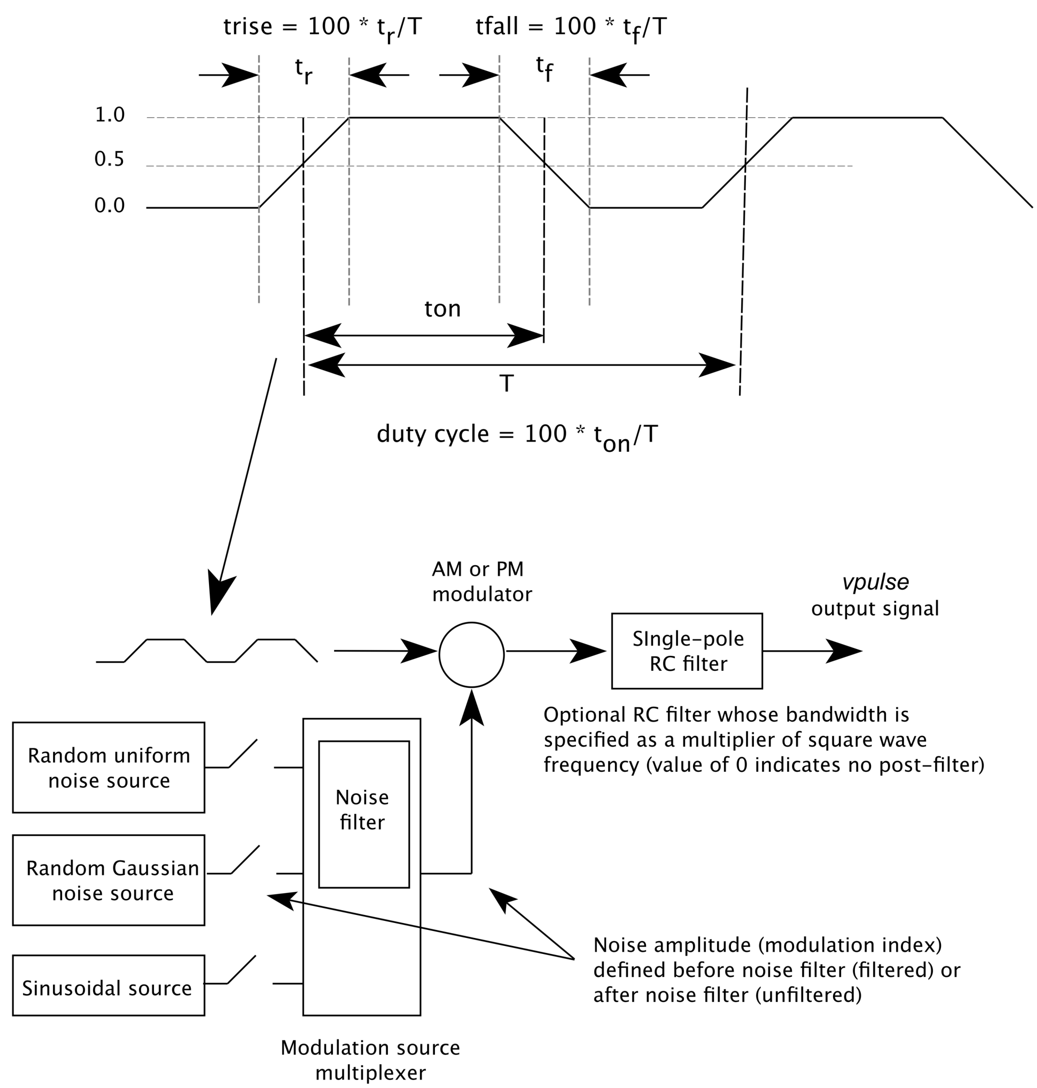
Section 2 provides a detailed description of the program and its installation. Several examples of its use are included in Section 3. Section 4 includes a summary of the program and its potential uses.

2. Program Description, Inputs, and Installation

2.1 Program Description

The custom C program, *vpulse*[1], generates periodic piecewise linear square waves. Figure 1 provides the definitions of its input waveform parameters and provides a block diagram that outlines its basic operation and optional features. The program features include the ability to specify an amount of amplitude or phase modulation of various types and to include a first-order lowpass post-filter to limit the bandwidth of the piecewise linear square wave.

Figure 1
Parameter Definitions and Block Diagram of *vpulse* Square Wave Signal Generator



---

[1] A UNIX version of the C-program *vpulse* with makefile and instructions is available from the author at the URL in reference [3].

The output of the program is a comma-separated variable file containing the samples of the simulated waveform with a single header line. Each line following the header consists of the sample time in seconds followed by the square wave sample value. The square wave amplitude has a logic high level of 1.0 with a logic low value of 0 and can be scaled to the desired amplitude.

The square wave frequency, initial phase, rise time, fall time, and duty cycle are programmable as are the number of periods to simulate, the number of samples per period, and number of periods to include in the output file.

Optionally, the square wave can be amplitude or phase modulated with a sinusoidal source or one of two random noise sources. The amount of modulation to apply is specified using its modulation index. The two random noise sources are followed by a single-pole filter with a programmable bandwidth to allow the use of bandlimited noise. If sinusoidal modulation is selected, the parameter used to specify the bandwidth of the noise filter is the frequency of the sinusoidal modulation. Sinusoidal amplitude or phase modulation is not filtered.

An optional first-order post-filter is available to limit the bandwidth of the square wave output and remove the discontinuities of a piecewise linear source.

The UNIX version of the program is invoked from the command line. If not invoked in its command line form with the correct number of inputs, *vpulse* will prompt for all inputs. In its simplest form, to generate 250 periods of a square wave at 125 MHz with an initial phase of 45 degrees, transition times of 20%, a duty cycle of 50%, no post filter, and 100 samples per period, the command line for *vpulse i*s:

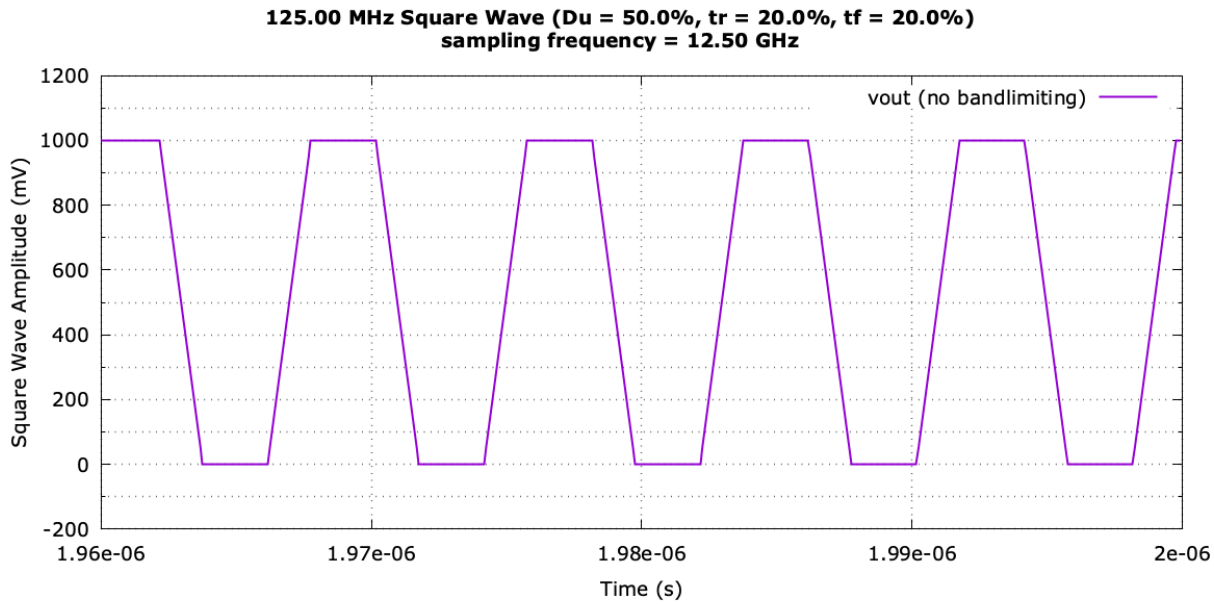
$ vpulse 125e6 45 20 20 50 0 100 250 250 Example command line [1]

The output comma-separated variable file is compressed if it exceeds 10K lines and has a filename containing a time stamp and descriptive information. In the example command line [1] shown above, the uncompressed filename is:

square_wave_125_MHz_ttran_rise_20_fall_20_percent_du_50_<timestamp>.csv

If **Gnuplot**[©] is installed and in the search path, the program will plot the last 5 cycles of the square wave and place the plot in a Portable Network Graphics (png) file with the same name as the comma-separated variable file but an extension of “png”. The plot showing the last five periods of the square wave generated by Example command line [1] is shown in Figure 2.

Figure 2



vpulse will invoke a power spectral density program and a jitter analysis program if either is in the executable path of the host system.

The power spectral density program, *psd_sppowr*, computes and plots the power spectral density of a waveform with N samples using last M samples where M is the largest binary multiple of samples computed from Equation [1]. The power spectral density is computed using 4 non overlapping segments and makes use of a Blackman Hanning window to minimize spectral leakage.

$$M = floor\left[\frac{\log 10(N)}{\log 10(2)}\right] \qquad \text{Equation [1]}$$

The power spectral density is plotted and saved to a file in Portable Network Graphics format if Gnuplot is installed and in the executable search path.

A jitter analysis program, *jitterhist*, computes the time interval error of the N time samples and the phase noise using the last M samples. It reports the peak-to-peak time interval error values for both the positive and negative edge transitions. It creates a file of the sample times with the positive and negative time interval error values and a second file containing the phase noise for each edge. Both files are in comma-separated format. If Gnuplot is installed and contained in the executable path, the positive and negative time interval errors and phase noise are plotted and saved to files in Portable Network Graphics format.

The two programs and their documentation are available from the URL in references [4] and [5]. They are useful as standalone programs for a power spectral density or time interval error analyses of any data.

2.2 *vpulse* Program Inputs

Inputs to *vpulse* include the frequency of the square wave, its transition times, its duty cycle, an initial phase, the number of time points per period, the number of periods to simulate, the bandwidth multiplier of an optional single-pole post filter, an optional modulation index, the location at which the modulation amplitude is specified, the modulation type (sinusoidal, uniform random noise, or Gaussian random noise), the noise bandwidth, and the type of modulation (AM/PM/none). Table 1 lists the inputs and shows values used in an example that includes amplitude modulation.

If the output square wave has no modulation, only arguments one through nine are necessary and the next five arguments need not be entered

If a non-zero value for the modulation index is included as a tenth argument, the next five arguments must be included. Inputs fifteen and sixteen specify if a power spectral density (PSD) and a time interval error (TIE) analysis is performed. These two inputs are optional and if not included, the default condition is to perform both a PSD analysis and TIE analysis. When there are only nine inputs, inputs fifteen and sixteen become inputs ten and eleven.

Table 1
vpulse Input Parameters

	vpulse v2.18 Parameter		Value	Units	Comments
	1	Square wave input frequency	1.00E+08	Hz	
	2	Initial phase	10	degrees	
	3	Rise time	1	% of period	
	4	Fall time	30	% of period	
	5	Duty cycle	65	% of period	
	6	First order filter bandwidth	0		Bandwidth of RC filter following square wave expressed as a multiple of the square wave frequency
	7	Number of points per period	1000		
	8	Number of periods	1000		
	9	Number of periods to plot	1000		
optional	10	modulation index	0.25		index value between 0.0 and 1.0
	11	noise amplitude location	filtered		Defines if the noise amplitude is defined before the noise filter (filtered) or after the noise filter (unfiltered)
	12	noise type	uniform		uniform, gaussian, or sinusoidal
	13	noise filter bandwidth	1.00E+07	Hz	
	14	modulation type	AM		AM or PM
	15	Perform PSD analysis? (y/n)	yes		<yes/no> or <y/n> with default value of yes if neither is entered
	16	Perform TIE analysis? (y/n)	yes		

The input arguments are audited to determine if their values are sufficient to generate an accurate set of waveform samples. If the audit fails, the program exits with a suggestion to modify or correct the argument flagged by the audit.

The optional single-pole post filter bandwidth is specified using argument six “First order filter bandwidth”. The filter bandwidth is normalized to the square wave frequency. Hence, a value of 3 for a 100 MHz square wave corresponds to a post-filter with a bandwidth of 300 MHz. If the “first order filter bandwidth” is set to zero (as shown the example of Table 1), there is no post-filter applied to the piecewise linear square wave.

The output waveform is sampled at a time interval defined by the square wave input period and the number of points per waveform specified by argument seven. It is simulated for the number of periods specified in argument eight. Argument nine, “Number of periods to plot”, truncates the simulated data and produces an output file with the “Number of periods to plot” in lieu of the entire “Number of periods”. In Table 1, the “Number of periods” and the “Number of periods to plot” are both set to 1000 and hence the output file will contain all 1000 periods of the simulation.

To modulate the square wave, a non-zero modulation index is specified as argument ten “modulation index”. In the case of sinusoidal or amplitude modulation, this expresses the amount of modulation as a fraction of the amplitude of the square wave. For phase modulation, the modulation index represents the phase modulation as a fraction of a unit interval. Hence the peak-to-peak of the amplitude or phase modulation is twice the modulation index. When modulating by random uniform noise, the maximum and minimum values of the zero mean noise are equal in absolute value to the modulation index. If the modulation source is Gaussian random noise, the modulation index is set as 3σ of the distribution, and hence twice the modulation index corresponds to 6σ of the distribution.

Argument eleven specifies the location at which the modulation index is specified. If the location is set to "filtered", the modulation index of the input to the noise filter is set to the specified modulation index. If the argument is set to "unfiltered", the modulation index of the filtered noise is set to the specified modulation index. The latter option sets the modulation index at the input of the noise filter to greater or equal to the specified modulation index.

For both the random uniform and random Gaussian noise generators, the seed is set to a value based on program run time and will vary from one simulation to the next. As a result, one should not expect the noise distributions from two modulated noise simulations using the same set of arguments to be identical.

The bandwidth of a first-order filter is specified as argument thirteen for both the random uniform and random Gaussian noise sources. If the noise type is set to sinusoidal, argument thirteen is the frequency of the sinusoidal modulating source. There is no bandwidth limiting filter in the modulation path when sinusoidal modulation is selected.

2.3 Program Outputs

In the absence of the power spectral density and the jitter analysis programs, *vpulse* creates an output file in comma-separated variable format with a single header line. The output filename is derived from the input values supplied to *vpulse* and is appended with a time stamp. The header line for the square wave includes the time constant and bandwidth of the post-filter specified as a *vpulse* input.

The first column contains the set of sample times in seconds, and the second column contains the corresponding normalized square wave amplitude. The square wave logic low value is zero and its logic high value is 1.0. If the total number of samples exceeds 10K, the output waveform file is compressed to save disk space and may be restored to its comma-separated variable format using the command shown in Example command [2].

```
$ gzip -d <compressed_filename.gz>
```
Example command line [2]

Using the *vpulse* command line shown in Example command line [1] produces a comma-separated variable file whose first six lines are shown in Table 2 where the intervening commas are not displayed.

If Gnuplot is installed and contained in the UNIX path, the program will also generate a Portable Network Graphics file containing up to the last five periods of the square wave. The name of the Portable Network Graphics file is identical to that of the comma-separated variable filename with its time stamp but uses a “png” extension in lieu of a “csv” extension.

Table 2
First 6 lines of *vpulse* comma-separated variable output file from
Example command line [1]

Time (s)	vout (no bandlimiting)
0.000000000000e+00	1.000000000000e+00
8.000000000000e-11	1.000000000000e+00
1.600000000000e-10	1.000000000000e+00
2.400000000000e-10	1.000000000000e+00
3.200000000000e-10	1.000000000000e+00

2.4 Program Installation

Version 2.21 of the program is available in the compressed tar file “vpulse_v2p21_031825.tar.gz” and may be downloaded from reference [3].

To install the program and create the executable “*vpulse*”, enter Example command line [3] in the directory in which you wish to locate the program.

```
$ tar -xvzf vpulse_v2p21_030925.tar.gz
```
Example command line [3]

This will create a directory “vpulse_v2p21” and extract the following directory structure:

Documentation/	example/	plotting_routines/
README.txt	include/	src/

Navigate to the “src” subdirectory, and issue the following two UNIX commands:

```
$ make
```
Example command line [4]

```
$ make clean
```
Example command line [5]

Issuing these two commands will create the executable “*vpulse*” and delete object files no longer needed. In addition, Example command line [4] will attempt to create a symbolic link to *vpulse* in your $HOME/bin directory if this directory exists. Assuming your $HOME/bin directory is contained in your executable search path (UNIX PATH variable), this will allow you to execute *vpulse* from any of your subdirectories using the command syntax provided in Example command line [1] in Section 2.1.

The example directory contains a sample run file "runline_example.sh" to execute program "*vpulse*." The most recent version of the documentation in Portable Document Format (pdf) is contained within the Documentation directory.

3. Examples of *vpulse* Output Waveforms

3.1 Description of Example Simulations

Six examples are provided to illustrate the operation and outputs produced by *vpulse* in combination with its optional power spectral density and jitter analysis components. Each example waveform has a frequency of 125 MHz with the same waveform transient parameters (transition times and duty cycle) and the same output post-filter bandwidth multiplier of 5 (625 MHz). However, the modulation type varies examples 1 and 2 are modulated using uniform random noise; examples 3 and 4 are modulated with Gaussian random noise; and examples 5 and 6 are sinusoidally modulated The modulation index for each of the six examples is 10% (0.10). The noise bandwidth chosen for examples 1 through 4 is 1 GHz, and the modulation frequency for examples 5 and 6 is 70/3 MHz or 23.3333 MHz. For each of the three modulation types, both an amplitude and phase modulation case are simulated. Each simulation includes 1025 periods and is uniformly sampled 1000 times per period for a total of 1.025e6 time samples.

A summary of the six cases and their respective input parameter values is provided in Table 3.

Table 3
Detailed *vpulse* Input Parameter Values for Each Example Simulation

	vpulse v2.0 Parameter	Example 1	Example 2	Example 3	Example 4	Example 5	Example 6	Units
1	Square wave input frequency	1.250E+08	1.250E+08	1.250E+08	1.250E+08	1.250E+08	1.250E+08	Hz
2	Initial phase	0	0	0	0	0	0	degrees
3	Rise time	10	10	10	10	10	10	% of period
4	Fall time	30	30	30	30	30	30	% of period
5	Duty cycle	24	24	24	24	24	24	% of period
6	First order filter bandwidth	5	5	5	5	5	5	
7	Number of points per period	1000	1000	1000	1000	1000	1000	
8	Number of periods	1025	1025	1025	1025	1025	1025	
9	Number of periods to plot	1025	1025	1025	1025	1025	1025	
10	modulation index	0.10	0.10	0.10	0.10	0.10	0.10	
11	noise amplitude location	filtered	filtered	filtered	filtered	filtered	filtered	
12	noise type	uniform	uniform	Gaussian	Gaussian	sinusoidal	sinusoidal	
13	noise filter bandwidth	1.00E+09	1.00E+09	1.00E+09	1.00E+09	2.3333E+07	2.3333E+07	Hz
14	modulation type	AM	PM	AN	PM	AM	PM	

3.2 Temporal and Fourier Components of Random Noise Sources

Examples 1 through 4 are modulated with a uniform or a Gaussian random noise source. An illustration of the temporal distribution of the unfiltered and 1 GHz first-order lowpass filtered noise sources is provided in Figure 3 and Figure 4 respectively. Each distribution contains 1.025e6 samples and the y axes are normalized to allow a direct comparison of the two distributions. As expected, the range of the unfiltered random uniform and random Gaussian corresponds to a modulation index of 0.10. It is also evident that the 1 GHz first-order filtering process reduces the ranges of each distribution as well as the entire shape of the uniform noise distribution. This latter result might be expected as the occasional large changes in successive sample values of a uniform distribution appear as high frequency components and are attenuated by the filtering process.

The power spectral densities of the unfiltered noise distributions are compared in Figure 5. Figure 6 compares the power spectral densities of the 1 GHz filtered noise distributions. The rms value of the uniform noise distribution is larger than the rms value of the Gaussian distribution. This result is expected since there are a larger number of uniform samples whose magnitude is close to the modulation index of 0.10 than there are for a Gaussian noise distribution.

Figure 3

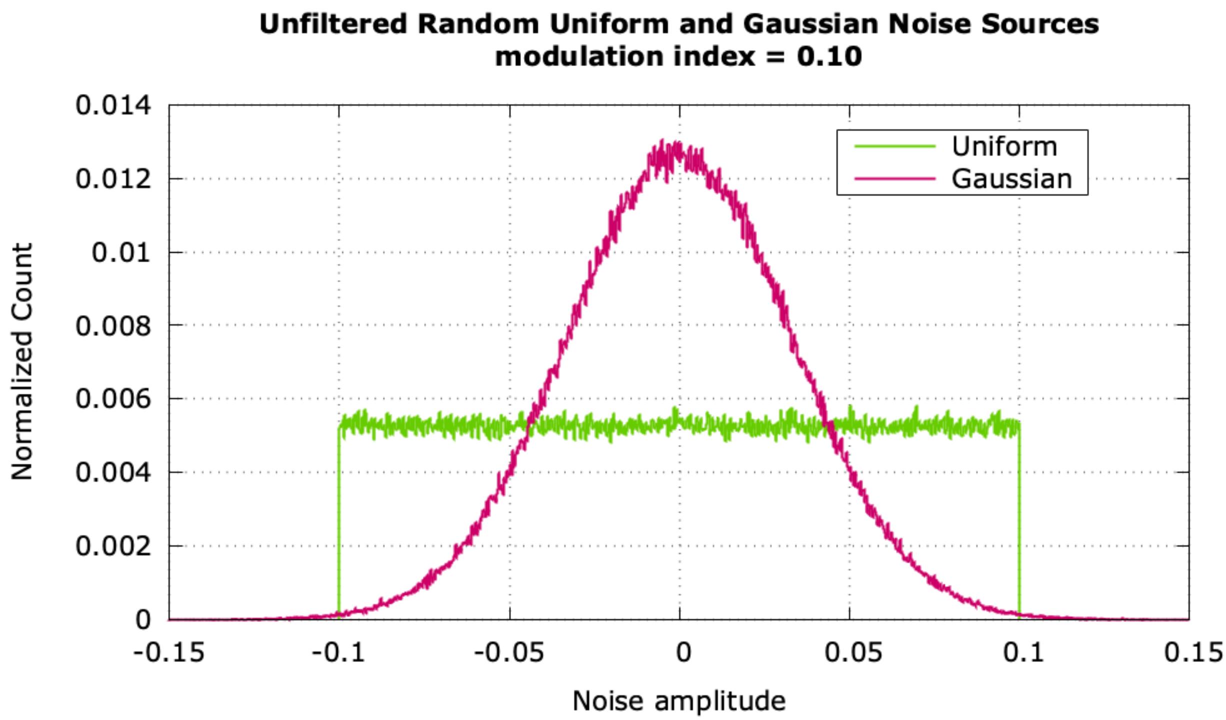


Figure 4

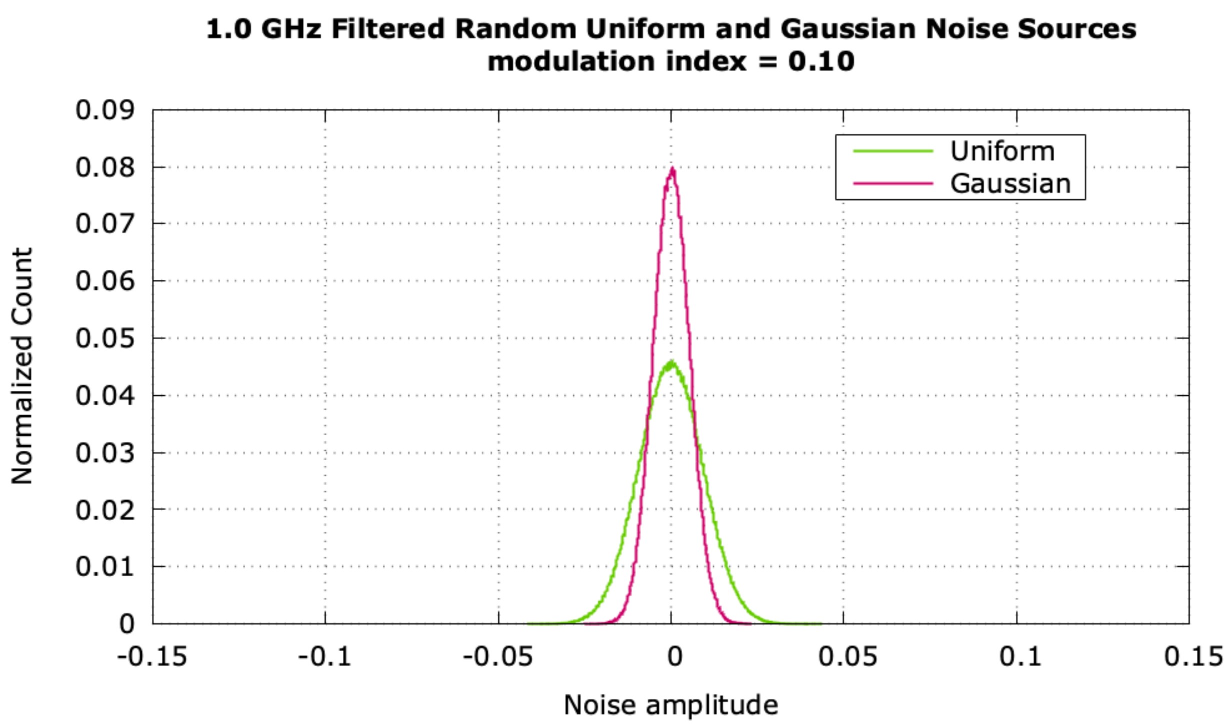


Figure 5

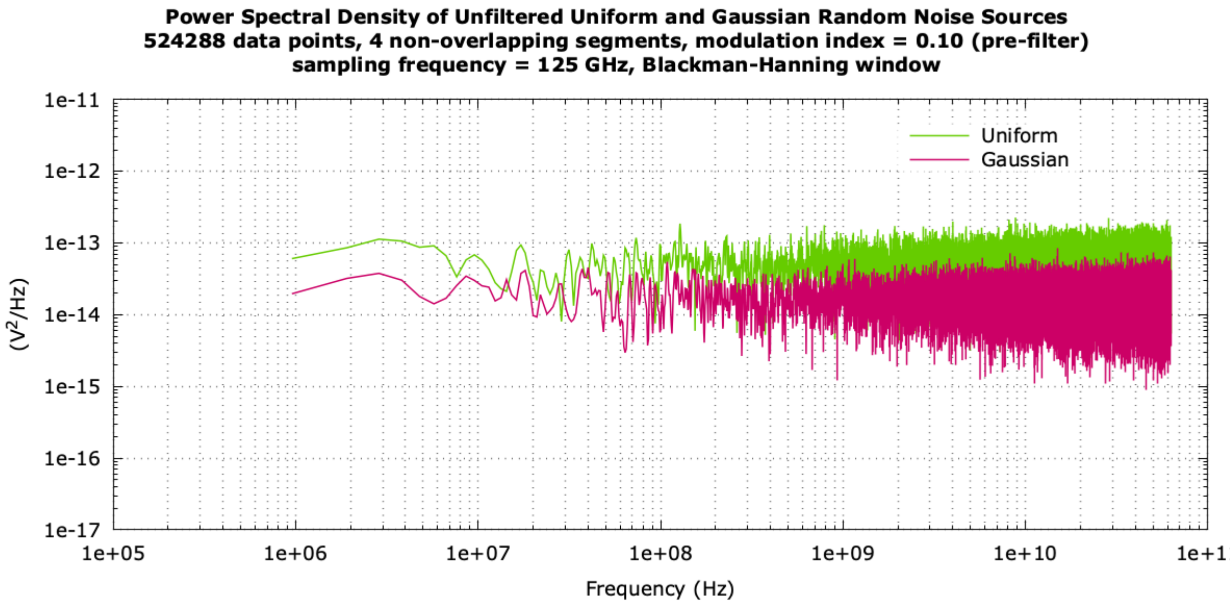
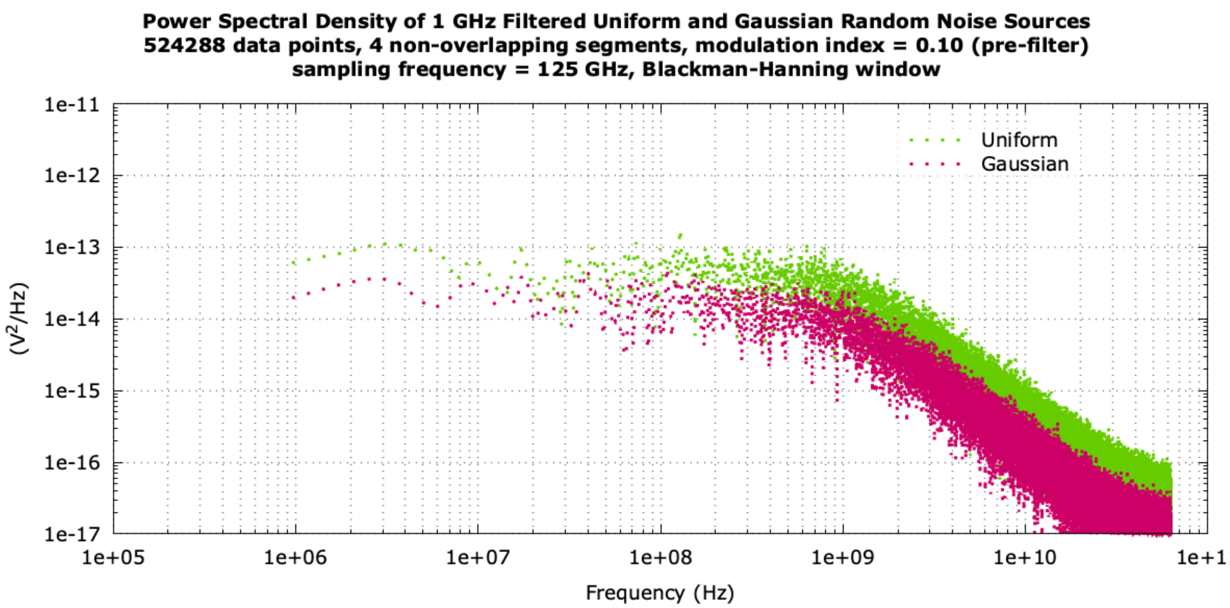


Figure 6



3.3 Random Noise Modulated Examples 1-4

3.3.1 Temporal and Fourier Components of Examples Modulated with Random Noise

In examples 1 through 4, *vpulse* generates 125 MHz square waves that are amplitude and phase modulated with a 1 GHz lowpass filtered uniform or Gaussian random noise source. In all cases, the modulation index is set to 0.10.

In the time domain, a comparison of the amplitude and phase modulated square waves is shown in Figure 7. In the time domain, it is difficult to distinguish the four waveforms unless the time axis is expanded. An expanded view of the four waveforms that more clearly distinguishes the four is provided in Figure 8. The transition times of the four waveforms are similar, but their amplitudes and phases are distinct. Figure 9 shows the measured transition times for the 125 MHz Gaussian noise modulated waveform. The measured transition times differ from the specified rise and fall times of 10% and 30% of the period respectively for two reasons: the 625 MHz bandwidth of first-order post filter limits the bandwidth of the piecewise linear waveform and impacts the transition times; the measured transition times use voltage thresholds of 10% (100 mV) and 90% (900 mV) and the specified transition times represent the time for the waveform to change from 0% (0 mV) to 100% (1000 mV).

Figure 7

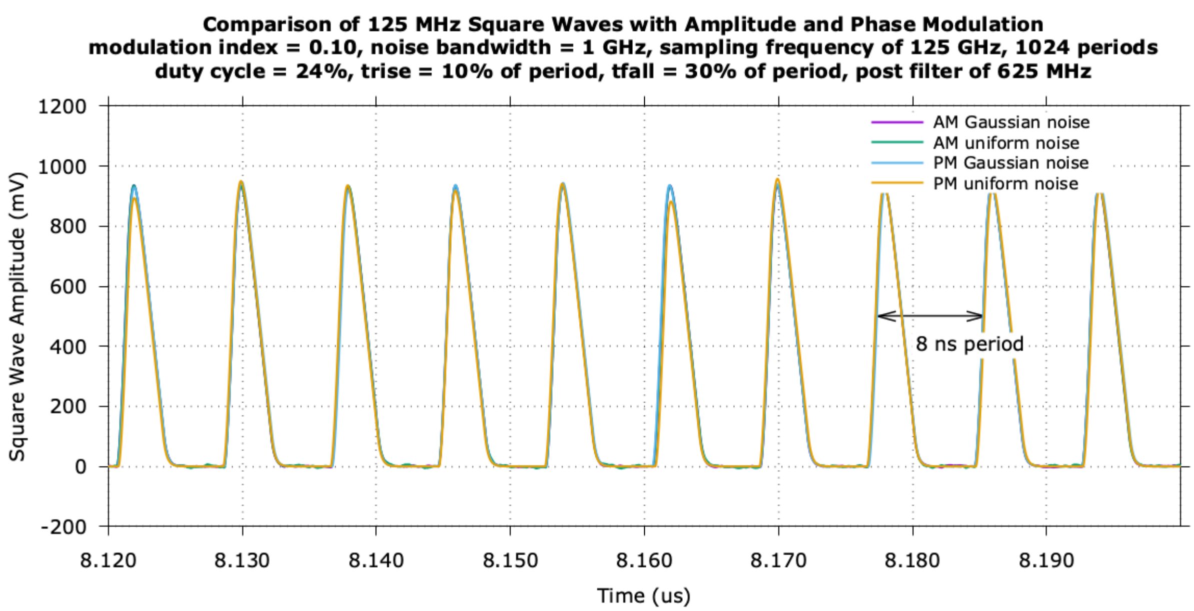


Figure 8

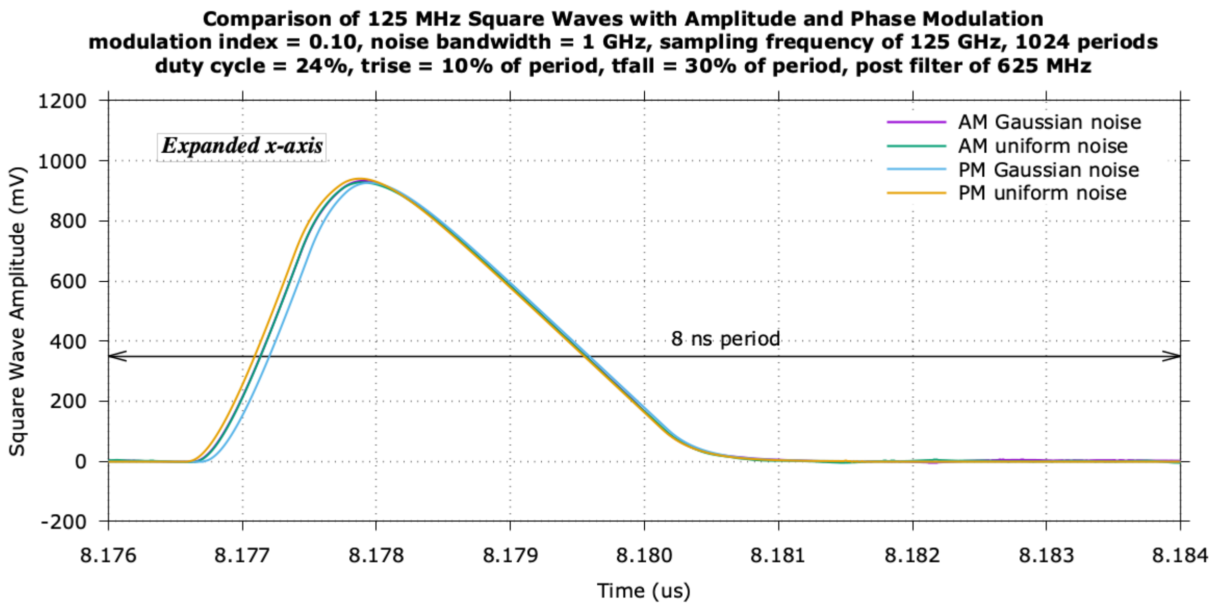
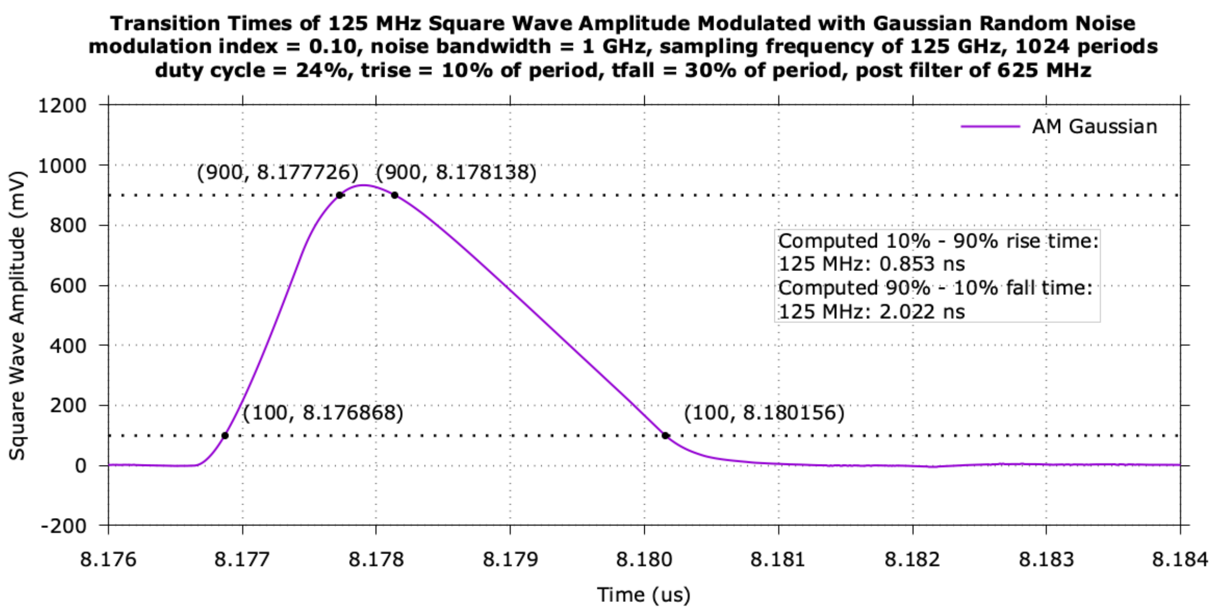


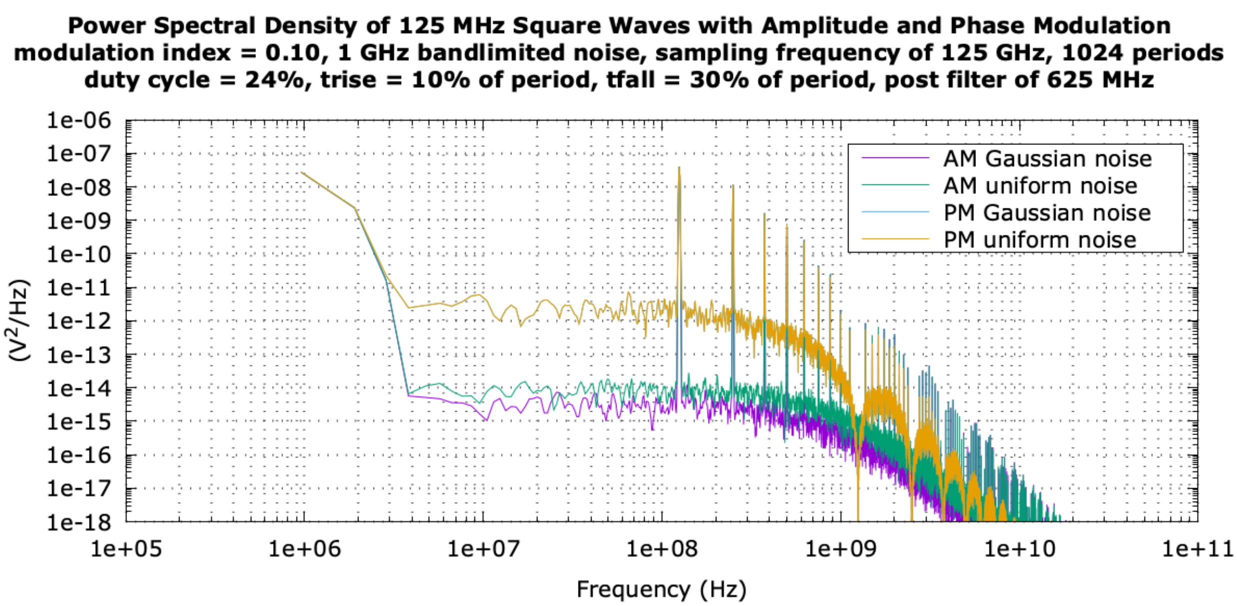
Figure 9



With the optional power spectral density analysis program *psd_sppowr* and Gnuplot installed and each in your executable path, *vpulse* will analyze the time domain samples and compute and plot the power spectral density of the simulated square wave using the last M time samples of the N waveform samples where M is the largest binary multiple that is less than or equal to N. The power spectral density is computed using 4 non-overlapping samples each of size M/4 with a Blackman-Hanning window to minimize spectral leakage.

Using this feature, Figure 10 compares the Fourier components of the amplitude and phase modulated square waves. Note that the two phase modulated examples have a much greater noise floor than the two amplitude modulate square waves. Also evident is the 1 GHz bandwidth of the noise sources as the noise floor for all waveforms is attenuated above the 1 GHz noise bandwidth. Although it was difficult to distinguish the four waveforms in the time domain, there is a pronounced difference between the noise floors of each waveform when viewed in the Fourier domain.

Figure 10



3.3.2 Time Interval Error (TIE) of Noise Modulated Examples

With the optional jitter analysis program *jitterhist* and Gnuplot installed and in the executable path, *vpulse* will analyze the time domain samples and compute and plot the positive and negative edge time interval error of the simulated square wave using a threshold of 500 mV.

For the amplitude modulated example, phase variation is induced by the amplitude modulation as it will impact the time location of each threshold crossing. The variation in the threshold crossing instant is a function of the transition time, and a longer transition time will lead to a greater amount of time interval error than a shorter transition time. Since the four examples have a larger fall time than rise time, it is expected that the negative edge TIE will exceed the positive edge TIE. Inspection of the amplitude modulated waveform time interval errors shown in Figure 11 does indeed indicate that the 0.21 mUI rms TIE measured using the positive edges is less than the negative edge rms TIE of 0.46 mUI.

The amount of phase modulation is specified as an input to *vpulse* as 0.10 (0.20 UIpp) and injected before the 1 GHz noise filter. Hence, the measured modulation index is expected to be less than 0.10. Figure 13 and Figure 14 detail the positive and negative edge TIE for the phase modulated waveforms. It is evident the peak-to-peak TIE values for both the positive and negative edges are less than 0.20 UIpp and consistent with the specified modulation index of 0.10.

Figure 11

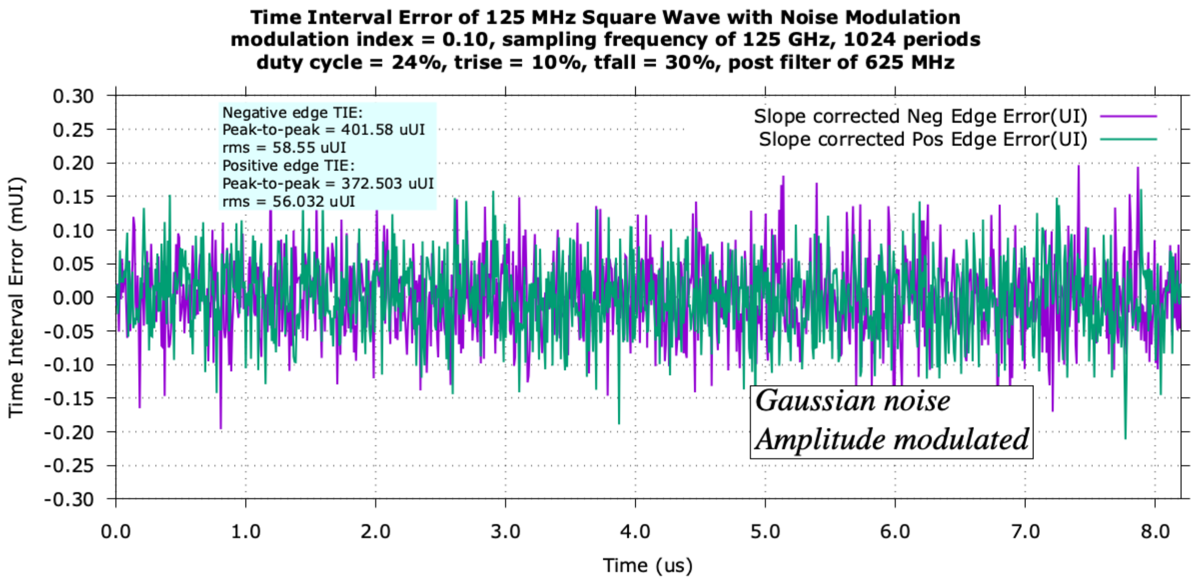


Figure 12

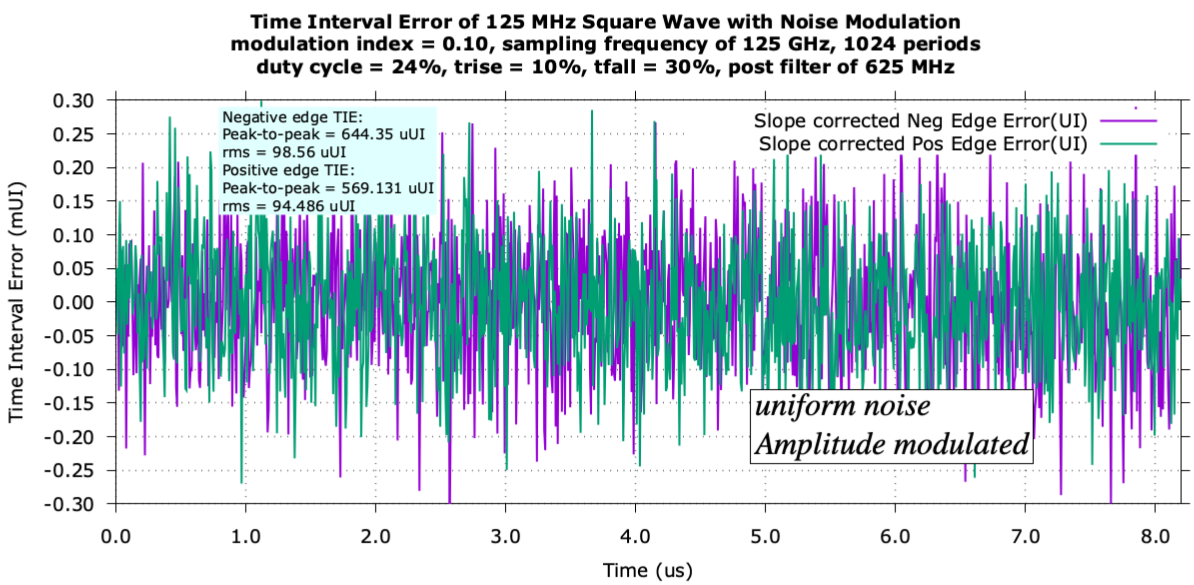


Figure 13

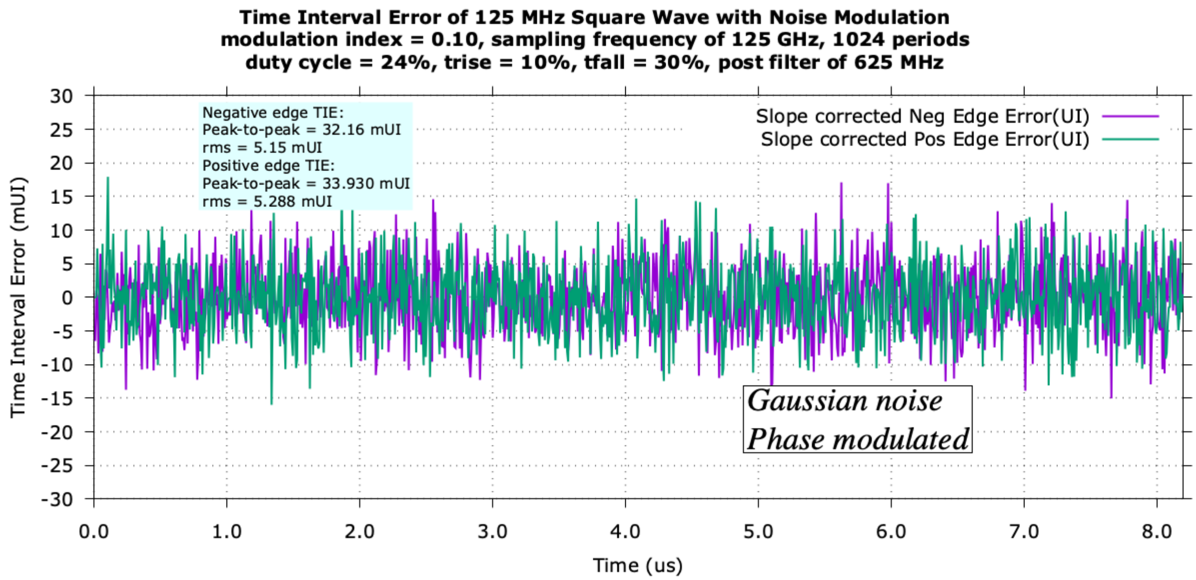


Figure 14

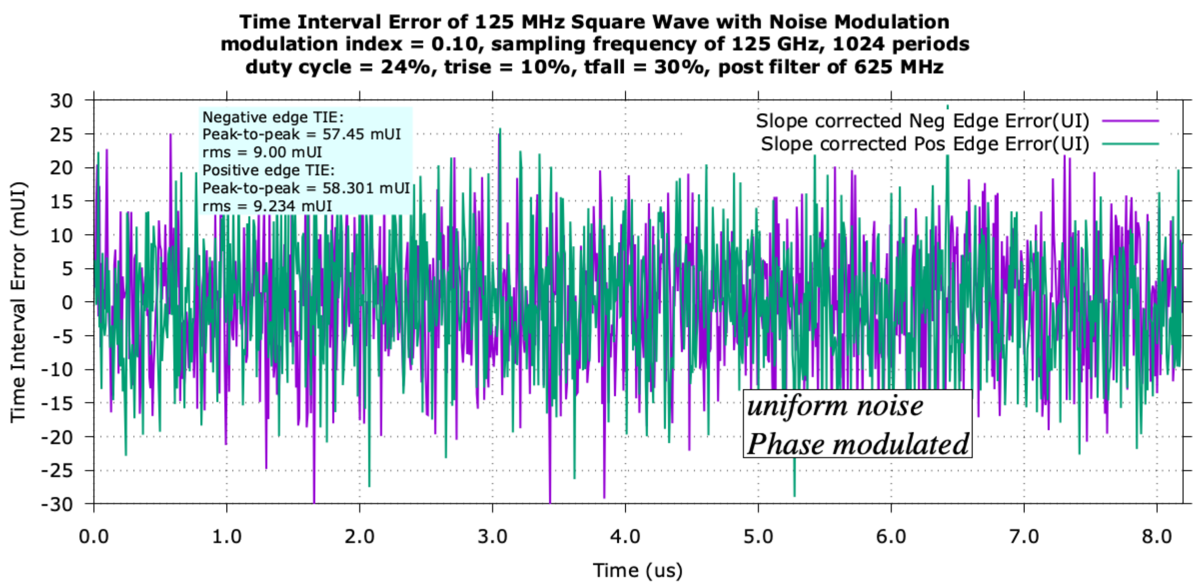


Figure 15 and Figure 16 illustrate the phase noise for the negative and positive edge based measurements respectively. Consistent with the time domain results, the TIE of the 125 MHz square waves with phase modulation exceeds the TIE of the amplitude modulated waveforms. For the same modulation type, the TIE of the uniform noise modulated square waves is greater than the TIE of the Gaussian noise modulated waveforms. The latter result is expected based on the difference in their probability distribution functions (see Figure 3 through Figure 6). The integrated phase noise rms jitter over the entire frequency range is within 0.10% of its respective time domain value.

Figure 15

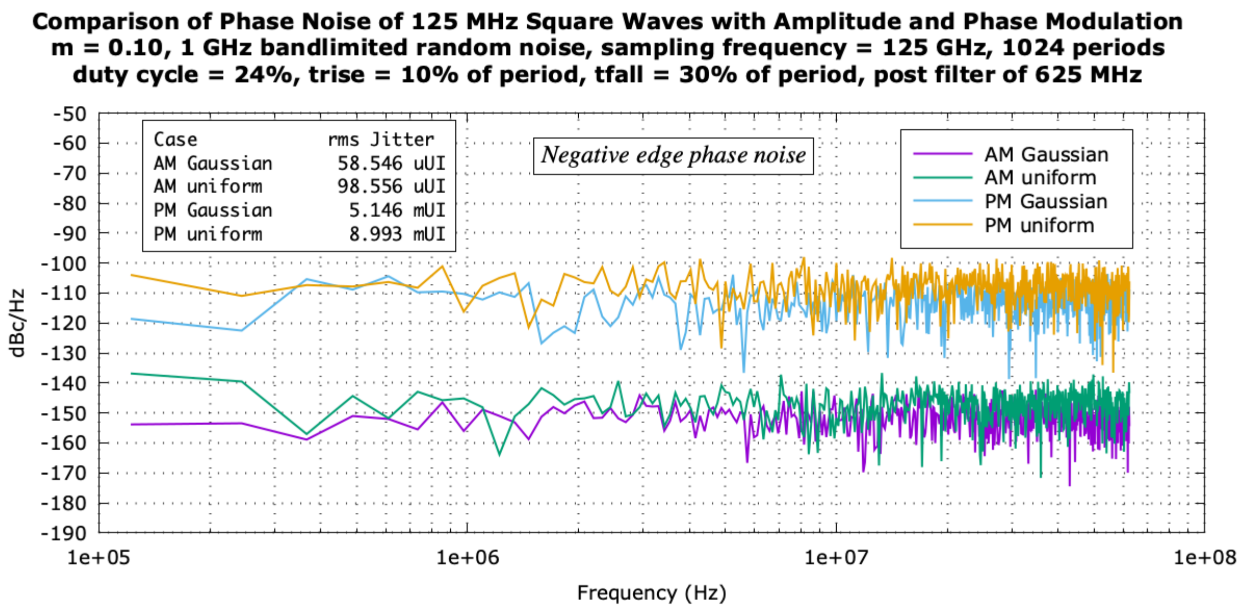
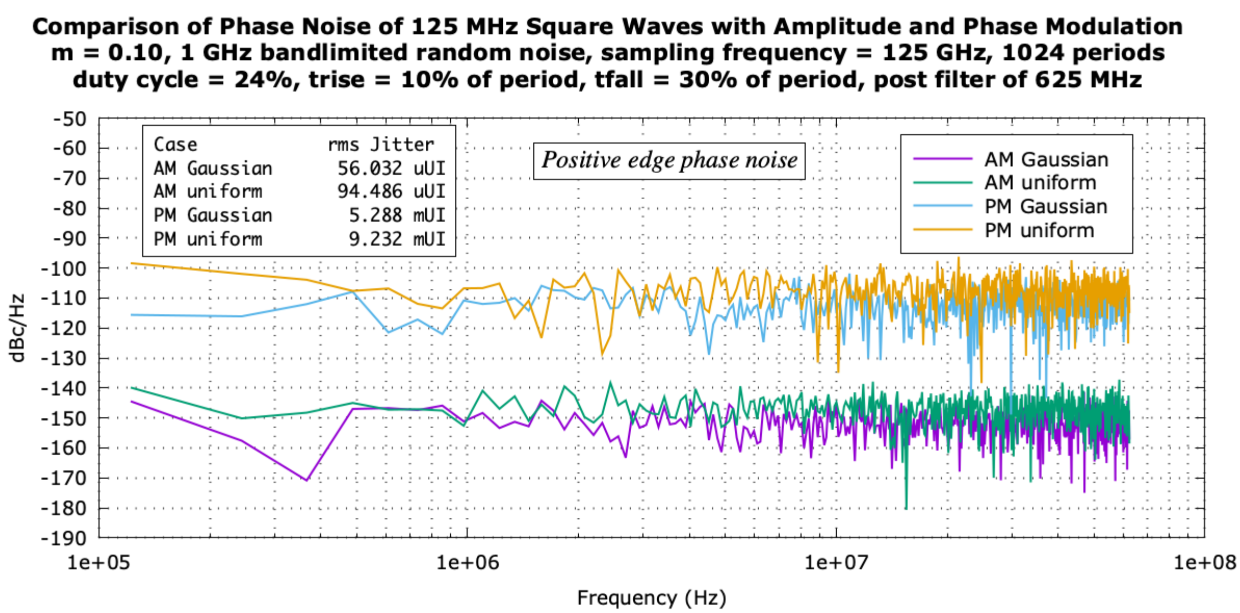


Figure 16



3.4 Sinusoidal Modulated Examples 5-6

3.4.1 Temporal and Fourier Components of Sinusoidal Modulated Examples

In examples 5 and 6, *vpulse* generates 125 MHz square waves with pure sinusoidal amplitude and phase modulation respectively. The modulation frequency of 70/3 MHz (23.3333 MHz) is chosen to include an non-integer number of 125 MHz periods.

In the time domain, a comparison of the amplitude and phase modulated square waves is shown in Figure 17 with an expanded view of the amplitude modulated waveform in Figure 18 with its measured 10% to 90% transition times. A close examination of the two waveforms in Figure 17 shows that the phase of the signal with phase modulation is varying periodically about the phase of the signal with amplitude modulation, and the peak amplitude of the amplitude modulated signal is varying periodically about the peak amplitude of the signal with phase modulation.

Figure 17

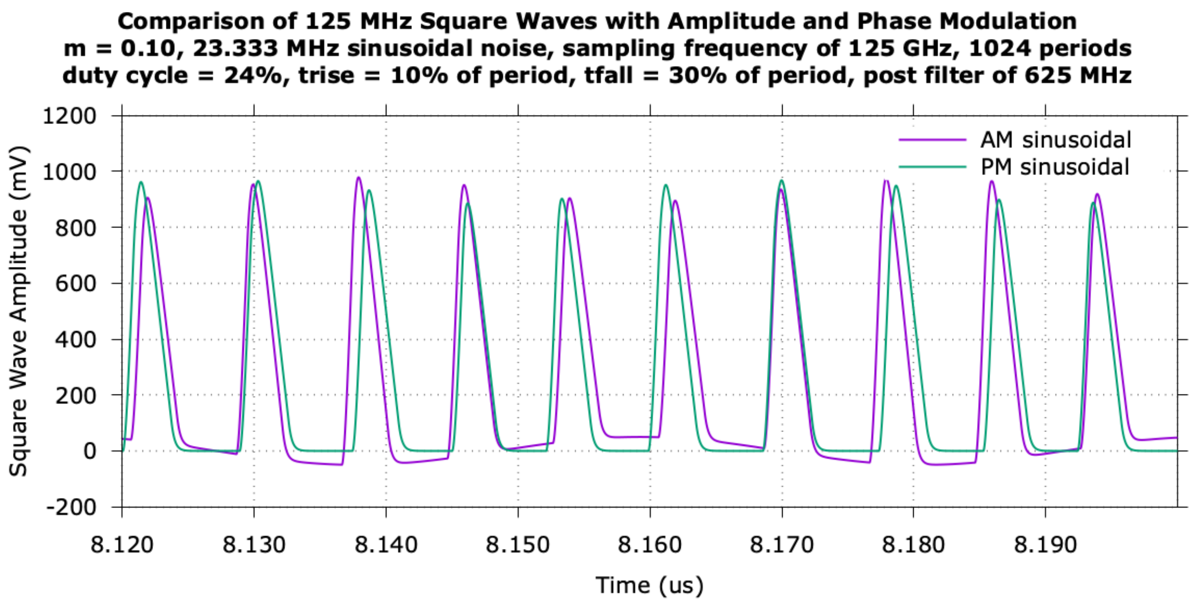
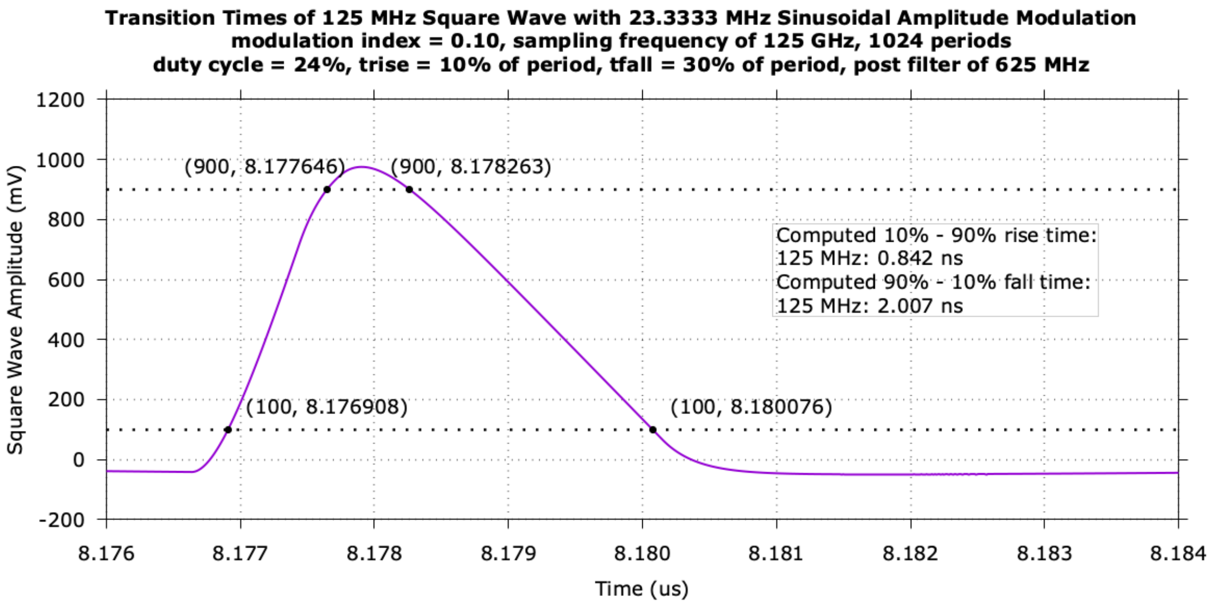


Figure 18



In the frequency domain, Figure 19 compares the power spectral density of the amplitude and phase modulated square waves.

The amplitude modulated signal produces a single upper and lower sideband 23.33 MHz from the harmonics of the 125 MHz square wave as shown in the expanded frequency axis plot of Figure 20. The tone at 23.33 MHz in Figure 20 is the upper sideband of the square-wave’s non-zero DC component.

Unlike the amplitude modulation process, phase modulation results in a series of sidebands about the 125 MHz waveform harmonics. The number of sidebands and power contained in the harmonics of 125 MHz are a function of the modulation index. In this example with a modulation index of 0.10, the 125 MHz harmonic power is relatively unchanged, but the number of 23.33 MHz sidebands exceeds the single upper and lower sideband seen in the amplitude modulation example.

Figure 19

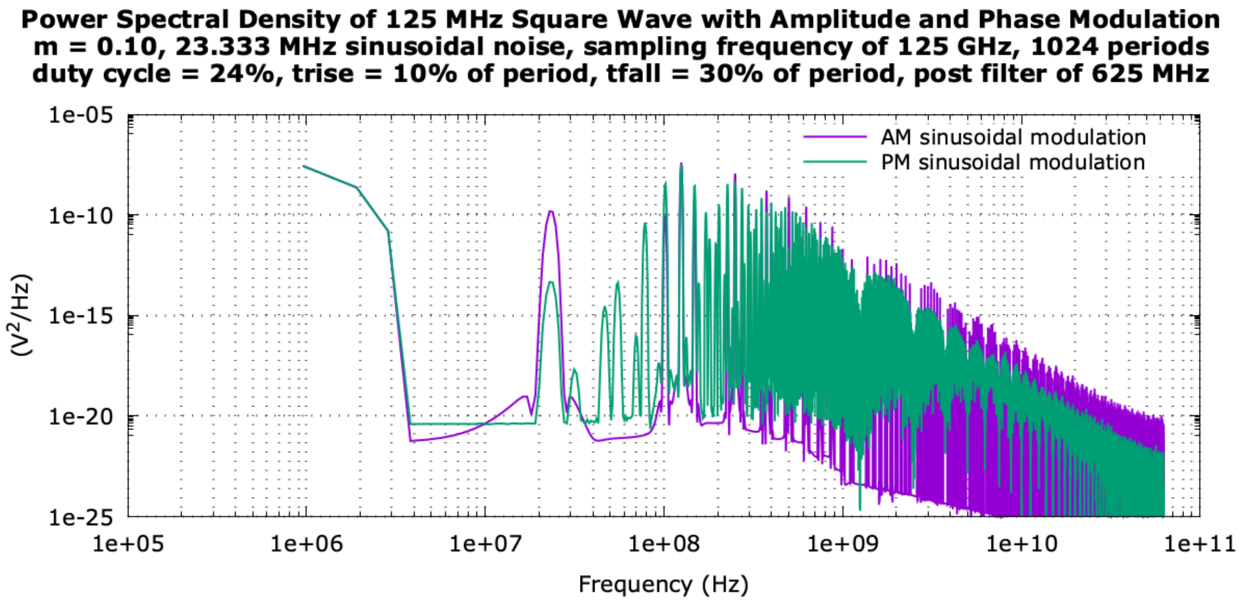
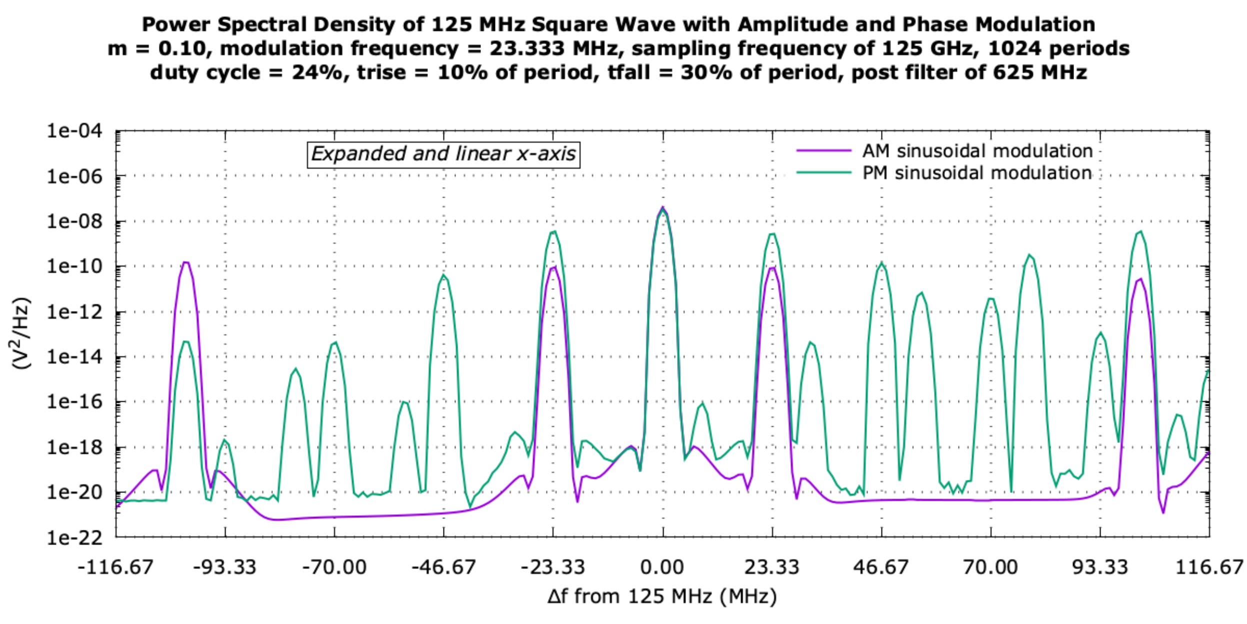
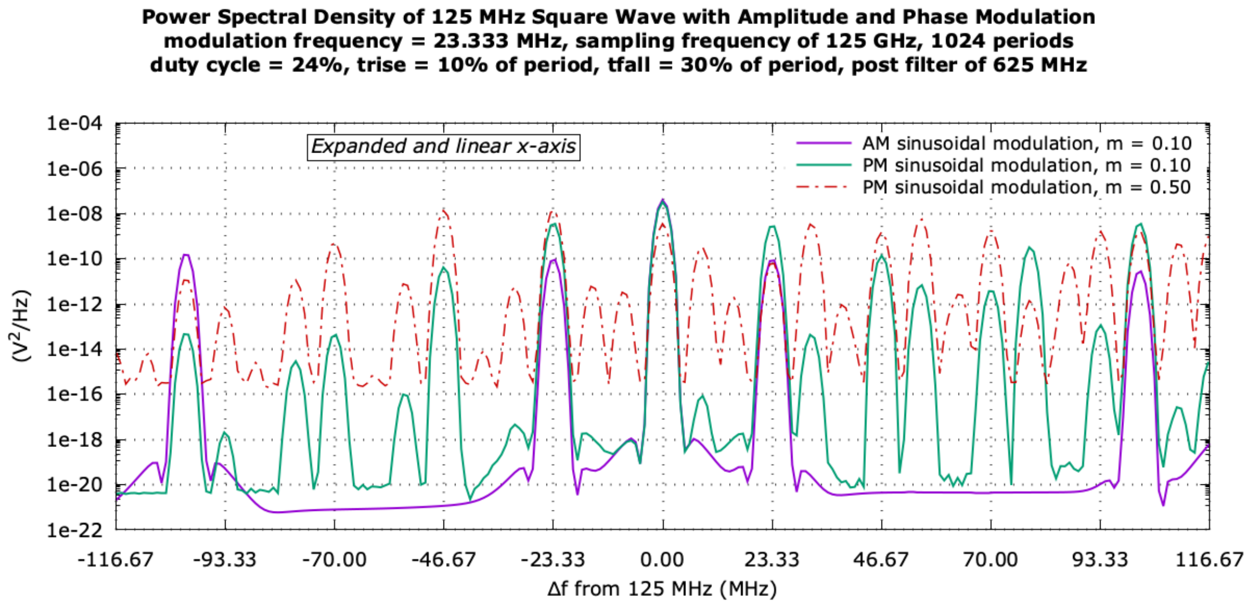


Figure 20



As the modulation index is increased from 0.10, the impact on the power of the 125 MHz harmonic and sideband power becomes more evident. In Figure 21, a comparison of the sideband and harmonic power for amplitude modulation with a modulation index of 0.10 and phase modulation with a modulation index of 0.10 and 0.50. When the phase modulation index is increased from 0.10 to 0.50, the power of the 125 MHz fundamental is reduced, and the number and magnitude of the 23.33 MHz sidebands are increased The observed sideband behavior and dependence on modulation index are consistent with well-known amplitude and phase modulation theory.

Figure 21



3.4.2 Time Interval Error (TIE) of Sinusoidal Modulated Examples

In the amplitude modulated example, phase variation is induced as the amplitude is modulated since it will impact the time of each 500 mV threshold crossing. The variation in the threshold crossing instant is a function of the slope of the transition time and the amount of amplitude variation. As a result of the greater waveform fall time, one might expect a proportionally greater amount of variation in negative edge threshold crossings than in positive edge threshold crossings as was observed for examples 1 and 3. Indeed, as in shown in the tabulated TIE results contained in Figure 22, the peak-to-peak negative edge TIE is greater than the peak-to-peak positive edge TIE.

The phase modulation index was specified as 0.10. With reference to Figure 23, the peak-to-peak TIE values for both the positive and negative edges are 0.20 and consistent with the specified modulation index.

Figure 24 and Figure 25 illustrate expanded x and y axis views of the amplitude and phase modulated TIE respectively.

Figure 22

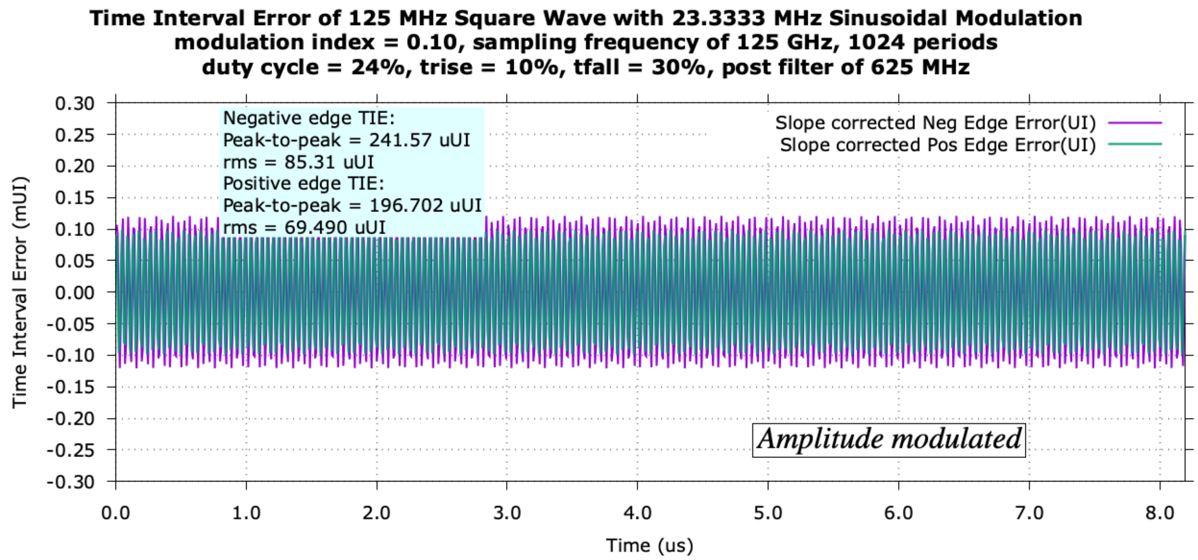


Figure 23

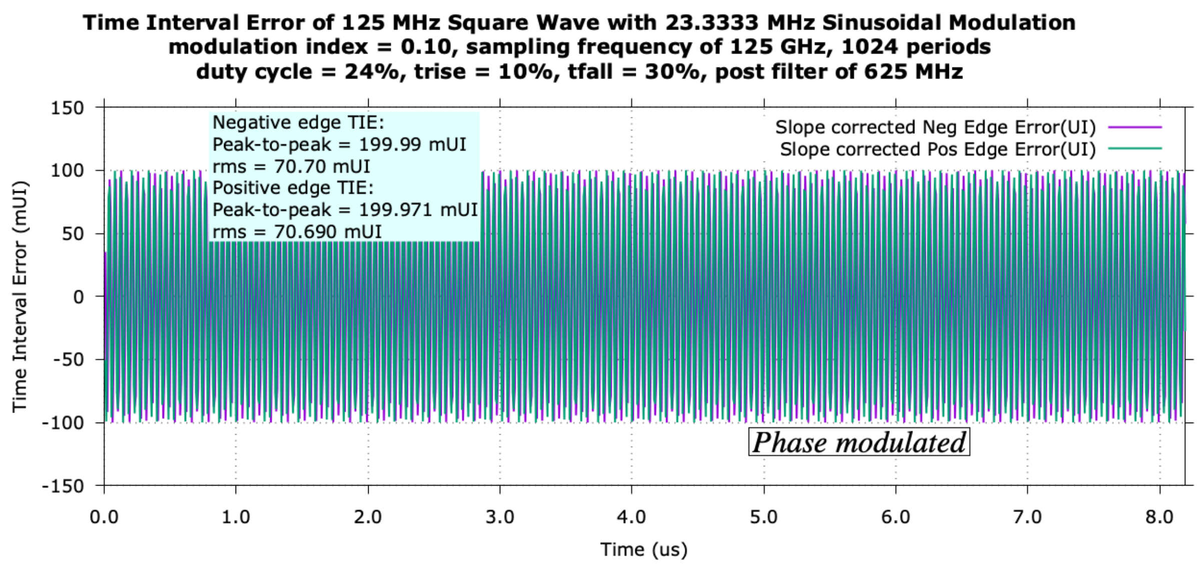


Figure 24

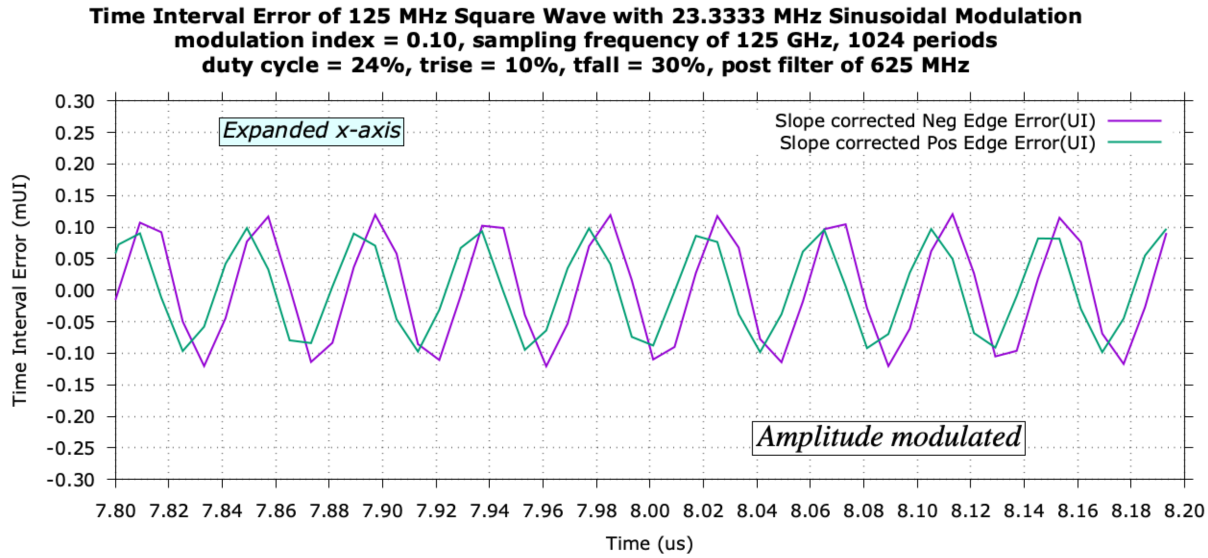
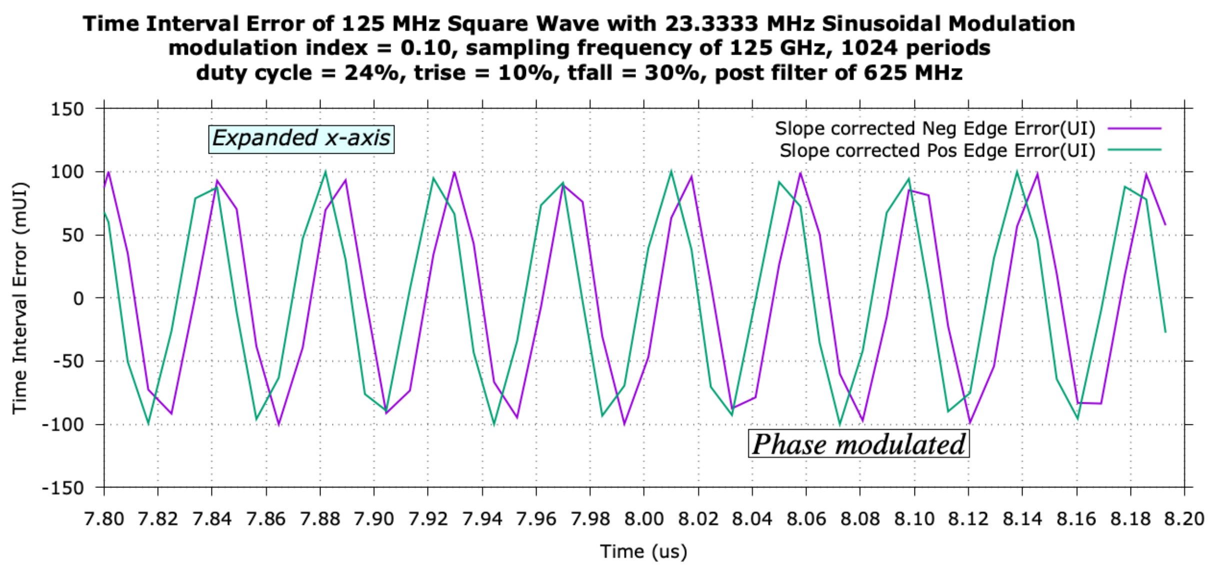
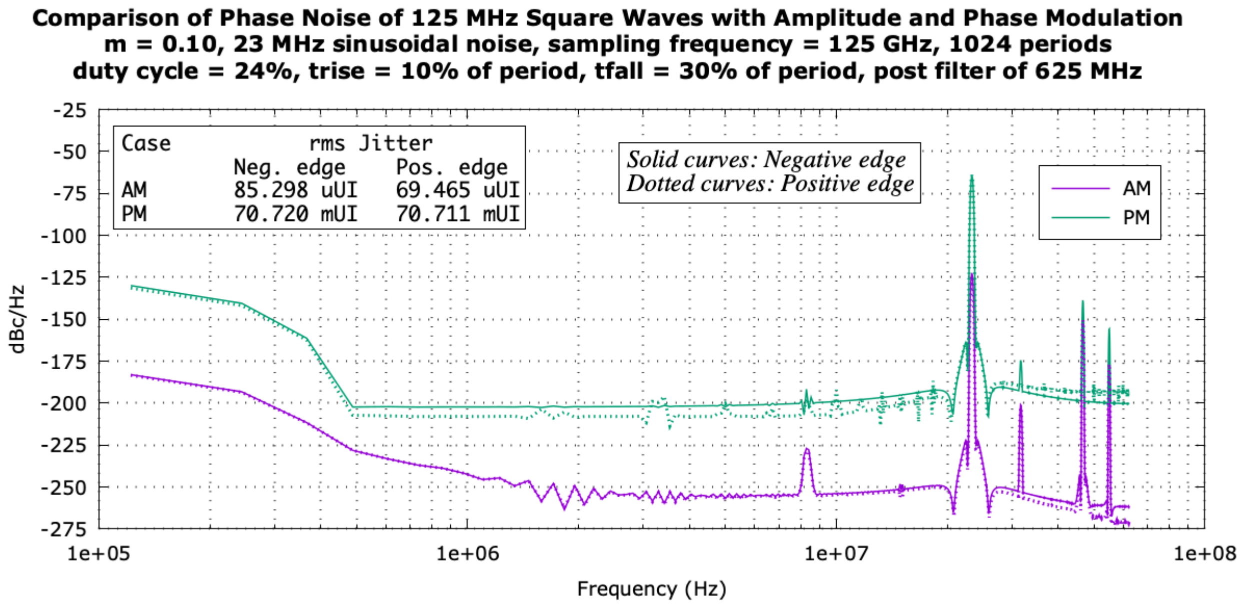


Figure 25



The results of a phase noise analysis are shown in Figure 26. Consistent with the time domain results, the peak amplitude of the phase modulated waveform time interval errors at 23.3333 MHz are about the same and both larger than those for the amplitude modulated waveform. For the amplitude modulated waveform, the rms value of the negative edge phase noise exceeds the rms of the positive edge.

Figure 26



4. Summary

Program *vpulse* generates a comma-separated variable file containing time domain samples of a square wave is described. In its simplest use case, the program generates a file containing samples of a piecewise linear square wave whose frequency and time domain waveform parameters are adjustable. Optionally, the piecewise linear square wave may be lowpass filtered with a single-pole filter.

The program also offers the ability to include an amount of amplitude or phase modulation by a sinusoidal, filtered random uniform, or filtered Gaussian noise source.

With **Gnuplot**© installed and in the executable path, the output waveform is displayed and saved in a graphics file.

Two post-processing modules, *psd_sppowr* and *jitterhist*, are offered for generating and viewing the power spectral density of the sampled square wave, its time interval error and phase noise. Each module is instantiated by *vpulse* if its executable is contained in the UNIX executable path variable. Both are available from the author as a standalone program at the URL in references [4] and [5].

The *vpulse* output file may be used as the piecewise linear source file to any SPICE based time domain simulator such as SPICE, Cadence Spectre®, or Synopsis HSPICE®. Alternately, the program and its two post-processing modules are useful for studying the properties of square waves in the temporal and Fourier domains.

Program *vpulse* is available for download at the URL in reference [3].

© Copyright (C) 1986 - 1993, 1998, 2004, 2007 Thomas Williams, Colin Kelley
http://www.gnuplot.info

References

[1] John W. Eaton, David Bateman, Søren Hauberg, Rik Wehbring (2023). GNU Octave version 8.2.0 manual: a high-level interactive language for numerical computations.
URL https://www.gnu.org/software/octave/doc/v8.2.0/

[2] The MathWorks Inc., Natick, Massachusetts: The MathWorks Inc.
https://www.mathworks.com

[3] https://1drv.ms/u/s!AnM-GsAEZPoSr22p4VYo2o08Rw-7?e=aLSg3N

[4] https://1drv.ms/u/c/12fa6404c01a3e73/QXM-GsAEZPoggBLAGAAAAAAAx5W5y2HjExOXwA

[5] https://1drv.ms/u/c/12fa6404c01a3e73/QXM-GsAEZPoggBIrGQAAAAAALR9oGth4jlVuoQ

Version History

v1.0 Initial release 1/2/2024

v1.01 1/2/2024

1. Updated directory structure in Section 2.4 to include README file and this document in portable document format
2. Updated document to reflect latest update to vpulse v2.07 1/2/2024Updated

v1.02 4/30/2024

1. Updated directory structure in Section 2.4 to include example and Documentation directories. Placed this version of the document in Portable Document Format into the Documentation directory. Moved the example runline file “"runline_example.sh"” into the example directory.
2. Updated Section 2.4 to reflect directory structure and installation instructions for vpulse v2.10.

v1.10 3/18/2025

1. Updated references to the version of *vpulse* in Section 2.4 to v2.21 with issue date of 3/18/2025.
2. Updated input parameters in Table 1 to reflect added logical inputs 15 and 16 to allow skipping power spectral density or TIE analyses. Re-wrote section 2.2 to reflect added inputs and their use.
3. Updated the number of periods and number of periods to plot in Table 3 from 1000 to 1025.
4. Updated all *vpulse* examples to use 1025 waveform periods in lieu of 1000 to increase resolution of Fourier analyses and . to reflect those of *vpulse* v2.20 or *vpulse* v2.21.
5. Updated all waveform figures to reflect those of *vpulse* v2.20 or *vpulse* v2.21.
6. Removed analysis of Fourier sideband amplitudes of TIE for sinusoidal modulated waveforms and analysis of the sideband amplitudes with duty cycle of the square wave. Section is no longer relevant with updates to *vpulse*.
7. Re-wrote summary in Section 4 and added URL for psd_sppowr and jitterhist compressed directories in references [4] and [5] respectively.
8. Updated Version History to include changes to prior document version 1.02.