



## Application Notes – Module\_01

### Waveform Formats

#### WAVEFORMS

A waveform is composed of a *waveform code*, *common parameters* and *waveform-specific parameters*. The waveform code is an index to select which waveform style should be used to generate the waveform contents. The common parameters are: *Delay*, *Data Length*, and Markers' *Start*, *Width*, *Polarity*, and *Enable*. The waveform-specific parameters are the parameters needed to characterize the selected waveforms.

The GUI/API provides some simple built-in waveforms. These built-in waveforms are parameterized by the waveform code, common parameters, and the waveform-specific parameters. To [select](#) a waveform in the GUI, the users can type in the waveform code or double click the waveform style textbox, which is on the right side of the waveform code, in the waveform tab. In the API, simply assign the property *code* by selecting a listed member from the enumeration class `WAVEFORM_CODE`.

The GUI comes with a set of generic built-in waveforms. The users can use these generic waveforms to generate new waveforms. In the waveform tab, all the parameters can be modified and [saved](#). The users can also use "save as" to create another waveform. For convenience, an alias waveform name can be used by the users to specify the customized waveforms.

In the GUI, the waveforms are stored in ASCII-based files with "wfa" or "wf" extensions. These wfa/wf files are only usable for the GUI operations. The files store the waveform parameters, such as waveform codes, delay, data length, markers' parameters, and the waveform-specific parameters. The users can use the GUI to modify these parameters and "save" it back to the file or "save as" to another new file. The waveforms provided with the GUI are generic and can be used to generate new waveforms for users' applications. If users use API to write their own application programs, they should not use the wfa/wf files since the files are only usable for the GUI. Instead, the users should use the [methods and properties](#) provided by the API to parameterize the waveforms.

**WAVEFORM CODE**

[Waveform Code](#) is an index number to specify the waveform style as listed in the following table. In the GUI, the users can double click the waveform style textbox and select an available waveform style from the dropdown list. In the API, an enumeration class, **WAVEFORM\_CODE**, can be used to select the waveform style.

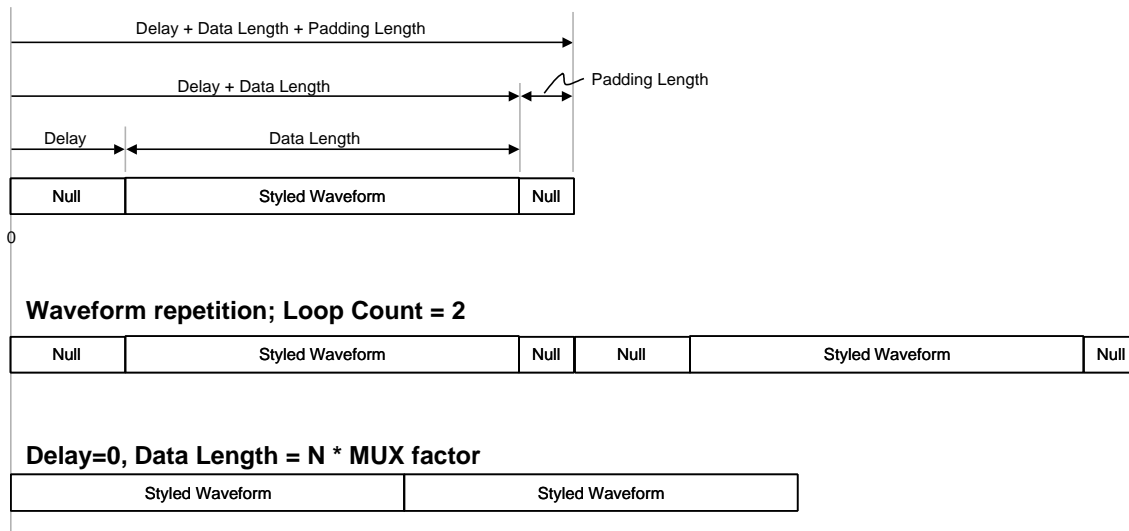
Waveform Code	AWG	DSM
0		Upward Chirping
1	Sine	Triangle Chirping (Up/Down)
2	Sine A/B	Downward Chirping
3	2 Tones	
4	N Tones	
5	Sine A/B Phased	
21	Chirp Phase Coherent	
22	Chirp Phase Continuous	
31	Ramp	
32	Sinc	
50	Pulse	
100	PNS	
256	User-Defined File	User-Define File

**COMMON PARAMETERS**

The *Delay* is used to offset the waveform in time. The *Data Length* is the length of the styled waveform in data sample. Due to the memory structure, the total length of the waveform is padded to the multiplexing (MUX) factor. The following table shows the multiplexing factors of the AWG and DSM modules.

Module	MUX Factor	Marker Sample Factor
DSM	4	1
AWG252	16	4
AWG272	16	4
AWG452	32	8
AWG472	32	8
AWG801	64	16

The length of a complete waveform is the sum of the Delay, Data Length, and Padding Length. The waveform repetition (specified in the Loop Count) is based on the complete waveform. The following figure shows the burst-mode waveform with Loop Count = 2. The Null data is 0 for the DSM and 0x800 for the AWG.



Markers are useful in synchronizing the waveforms. There is one marker available for the DSM and three markers available for the AWG. The properties of markers are *Start*, *Width*, *Polarity*, and *Enable*. The Marker 1 of the AWG has no *Polarity* and *Enable* properties since the marker is directly from the memory and there is no control circuit on it. The *Start* property specifies when the marker becomes active and the *Width* specifies how long the active marker lasts. However, because the markers are without multiplexing, the marker start and width are in multiple of the *Marker Sample Factor* as shown in the above table. For examples, marker2.start = 0x10 and marker2.width = 0x20: the marker will be active from 16<sup>th</sup> to 47<sup>th</sup> data sample for the DSM; and the marker 2 will be active from 64<sup>th</sup> to 191<sup>st</sup> data sample for the AWG252 and AWG272.

## WAVEFORM-SPECIFIC PARAMETERS (FOR AWG ONLY)

In the GUI, once the waveform code is selected, a corresponding table contents will be displayed. The table content consists of waveform-specific parameters for styling the waveform. These parameters vary for the different waveform codes, thus called waveform-specific parameters. The format and units of the parameters are labeled accordingly. The formats are in integer (Int), hexadecimal (Hex) and decimal (Dec). For example, in the AWG GUI, the waveform-specific parameters for the built-in PNS waveform can be specified as:

Parameter	Format	Value
Order (Int)	Int	15
Seed (Hex)	Hex	0
Tr, Fraction of To (Dec)	Dec	0.5
Tf, Fraction of To (Dec)	Dec	0.5
To (samples) (Int)	Int	10
Amp (full scale) (Dec)	Dec	1

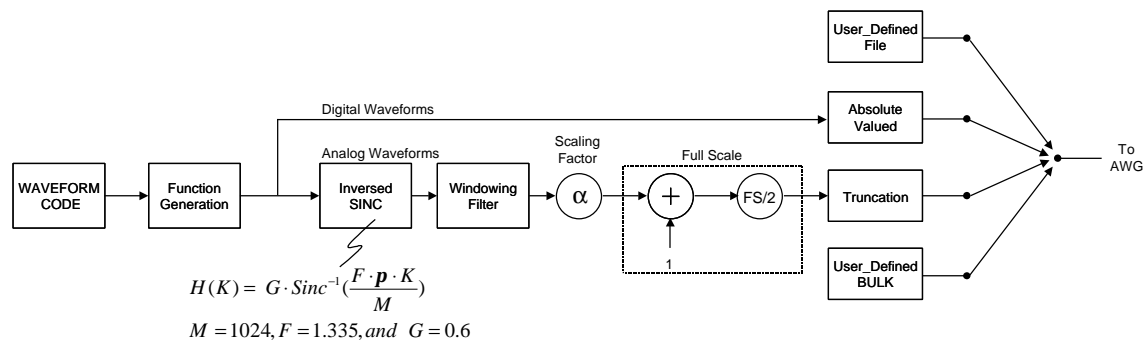
In the API applications, there are two groups of waveform-specific parameters, `waveform_parameter_ux` and `waveform_parameter_dx`, where  $x$  is 0~7, for the built-in waveforms. The `waveform_parameter_ux` are for the parameters in 'unsigned'. All the waveform-specific parameters with (Int) and (Hex) formats in the GUI are in 'unsigned' in the API and should be orderly specified in `waveform_parameter_ux`. The `waveform_parameter_dx` are for the parameters in 'double'. All the waveform-specific parameters with the (Dec) format in the GUI are in 'double' in the API and should be orderly specified in `waveform_parameter_dx`. The following code shows the waveform-specific parameters for the PNS built-in waveform.

```
//---- Set Waveform-specific Parameters
//---- Note that the parameters are in the same order as they appear in the GUI
//---- but they are enumerated starting from 0 for each type independently
//---- u (integer or hexadecimal)
//---- d (decimal)
awg.waveform_parameter_u0 = 15;      // Order      in unsigned
awg.waveform_parameter_u1 = 0x0;    // Seed      in unsigned
awg.waveform_parameter_d0 = 0.5;    // Tr        in double
awg.waveform_parameter_d1 = 0.5;    // Tf        in double
awg.waveform_parameter_u2 = 10;     // To        in unsigned
awg.waveform_parameter_d2 = 1;      // Amplitude in double
```

The parameters are in sequence and the index  $x$  is enumerated starting from 0. The complete waveform-specific parameters correspondence for the AWG can be found in the appendix, which is generated by an example code, `AWG_waveform_explorer.exe`. The source and the executable codes of the `AWG_waveform_explorer` are also included in the API example.

### WAVEFORM DATA FLOW AND PROCESS (FOR AWG ONLY)

The following figure shows the waveform generation and data flow for the AWG. There are two types of waveforms: Analog and Digital. The analog waveforms include sine, sinc, two tones, multi-tone, and chirpings. The digital waveforms are ramp, PNS, user-defined file, and user-defined bulk. The analog waveform data can use an [inversed SINC filter](#) to compensate the zero-order-hold DAC response. Moreover, windowing filters can be used to minimize the side lobes for non-periodic waveforms. The amplitude of the analog waveforms can be scaled relative to the full scale of the DAC. A truncation process is performed to convert the analog waveform data to the digital data. On the other hand, the digital waveforms contain the absolute valued data, which can be directly downloaded to the AWG. Both the user-defined waveforms are also digital data. The absolute valued data are directly bit-wise converted to the analog signal. If the digital data is greater than the full scale of the DAC, the overflowed digits will be ignored.



In the GUI, the selection of the inversed SINC function, windowing filter, and the scaling factor can be set by Option::Scaling/Filter menu item. In the API, properties `wv_scaling_factor`, `wv_full_scale`, `window` can be used to setup the filter processes for the analog built-in waveforms.

## USER-DEFINED FILE

In most cases, the desired waveforms are more complicate than the built-in waveforms. The GUI and API can accept customized waveforms from the user-defined files. The user-defined files are in ASCII format and should be with an extension of “uda” (for AWG) or “ud” (for DSM). The user-defined files can be located in different folders. To import the user-defined file in the GUI, the users need to:

1. specify the waveform *code*, 256, or WAVEFORM\_CODE::USER\_DEFINED
2. [open the user-defined file](#)
3. check if the number of points matched to the user-defined file
4. if #type=1, the markers' properties should be filled in
5. **download** the user-defined file to the module

If the user-defined file is being modified by an external program, the **reload** button can be used to update the contents. The users still need to **download** the waveforms to the modules after using the **reload**. The **reload** only updates the contents in the API but not in the modules. It is the **download** command transferring the user-defined waveforms to the module. The customized waveform can be “saved” or “saved as” just like the built-in waveforms. In the next time, when the customized waveform is selected, the location of the user-defined file and the common parameters are automatically recalled.

The user-defined file is in ASCII with a line comment character ';'. The user-defined file is composed of a control section and a data section. In the control section, two controls are used: "#type" and "#hex". The *type* is a 32-bit bit-wise control word. The *type* control is to specify the contents of the data section. The bit 0 of the *type* indicates the data in the data section is an amplitude word for the AWG and frequency word for the DSM. The AWG users should always set this bit to '1'. The bit 1 of the *type* indicates the data in the data section is an absolute frequency for the DSM and should be used only for the DSM. The bit 2 of the *type* indicates if the data section contains the marker controls or not. If the bit 2 of the *type* is set '0', the data section contains only one-column data section. If the bit 2 of the *type* is set '1', the data section contains two columns data section, where the first column is for the data and the second column is for the markers' controls. In summary for the *type* control:

<i>type</i> bit	Data Section Contents
0	Data Word
1	Absolute Frequency in (Hz) for the DSM
2	Markers' Controls Exist

The *hex* is a bit-wise control bit to indicate the data section is in hexadecimal ('1') or in decimal ('0') format.

In the data section, the first column is the word column. For the DSM, the word is 32-bit unsigned integer and for the AWG, the word is 12-bit unsigned integer. If "#type=1" is used, the data section is one column and only for the data word. The data section only

defines the data word but not the markers. The markers can be controlled by the markers's properties in the GUI or in the API.

To further control the markers, "#type=5" can be used and the data section has two columns. The first column is for the data word and the second column is for the markers' control(s). For the markers' controls, the three LSB's represent the three markers; the bit 0 is for marker 1, the bit 1 is for marker 2, and the bit 2 is for the marker 3. Similar to the GUI Markers' properties, the makers' controls are only sampled at multiple of *Marker Sample Factor* (*MUX factor/4*) in data section.

The data section is in a stream format; i.e. in timely sequence. For example, for an AWG with *type=1*:

```
; Control Section
#type=1      ; Single column format
#hex=1      ; Hexadecimal
;
; Data Section
000          ; at the first sample, amplitude = 0
004          ; at the second sample, amplitude = 4 (full scale is FFF)
008          ; at the third sample, amplitude = 8
00C          ; at the fourth sample, amplitude = 0xC (Hex format is used)
```

If *#type=5* is used, the data section are in two columns for both amplitude data and markers. The three-bit marker column represents 3 markers. For AWG272 and AWG252, the *Marker Sample Factor* is 4, therefore the marker are sampled every 4 samples. Markers controls not at the multiple of the *Marker Sample Factor* are ignored.

```
; Control Section
#type=5      ; Two column format
#hex=1      ; Hexadecimal
;
; Data Section
000  7      ; at the first sample, three marker are '1'
004  0      ; at the second sample, marker column is unused
008  0      ; at the third sample, marker column is unused
00C  0      ; at the forth sample, marker column is unused
010  3      ; at the fifth sample, marker 3 is '0' and marker 2 and 1 are '1'
014  0      ; marker column is unused
018  0      ; marker column is unused
01C  0      ; marker column is unused
```

For a DSM with *type=1*:

```
#type=1      ; one column format, Frequency Code
#hex=1
;
; Data Section
00100000    ; Frequency Code = 0x00100000
00200000    ; Frequency Code = 0x00200000
00300000    ; Frequency Code = 0x00300000
00400000    ; Frequency Code = 0x00400000
00500000    ; Frequency Code = 0x00500000
```

For a DSM with *type=2*:

```
#type=2      ; one column format, Absolute Frequency in Hz
#hex=0
;
; Data Section
1000000     ; Frequency = 1 MHz
2000000     ; Frequency = 2 MHz
3000000     ; Frequency = 3 MHz
4000000     ; Frequency = 4 MHz
5000000     ; Frequency = 5 MHz
6000000     ; Frequency = 6 MHz
7000000     ; Frequency = 7 MHz
```

For a DSM with *type=5*:

```
#type=1      ; two column format, Frequency Code & Marker
#hex=1
;
; Data Section
00100000    1      ; Frequency Code = 0x00100000, Marker = 1
00200000    0      ; Frequency Code = 0x00200000, Marker = 0
00400000    1      ; Frequency Code = 0x00400000, Marker = 1
00800000    1      ; Frequency Code = 0x00800000, Marker = 1
01000000    0      ; Frequency Code = 0x01000000, Marker = 0
```

For a DSM with *type=6*:

```
#type=6      ; two column format, Absolute Frequency in Hz
#hex=0
;
; Data Section
1000000     1      ; Frequency = 1 MHz, Marker = 1
2000000     1      ; Frequency = 2 MHz, Marker = 1
10000000    0      ; Frequency = 10 MHz, Marker = 0
20000000    0      ; Frequency = 20 MHz, Marker = 0
30000000    1      ; Frequency = 30 MHz, Marker = 1
```

## APPENDIX

The complete style name, waveform-specific parameters of the built-in waveforms for the AWG are listed in the following. The list is generated by the example code, *AWG\_Waveform\_Explorer.exe*. To use newly added built-in waveforms in AWG, it's recommended the users run the *Waveform\_Explorer.exe* at each API update.

Code= 1	Style: Sine Parameter number: 1	parameter[0]: Freq. (Hz) (Dec) (d)	waveform_parameter_d0
Code= 2	Style: Sine A/B Parameter number: 2	parameter[0]: DIV A (Hex) (u) parameter[1]: DIV B (Hex) (u)	waveform_parameter_u0 waveform_parameter_u1
Code= 3	Style: Two Tones A/B + C/D Parameter number: 4	parameter[0]: DIV A (Hex) (u) parameter[1]: DIV B (Hex) (u) parameter[2]: DIV C (Hex) (u) parameter[3]: DIV D (Hex) (u)	waveform_parameter_u0 waveform_parameter_u1 waveform_parameter_u2 waveform_parameter_u3
Code= 4	Style: N Tones Parameter number: 3	parameter[0]: Freq (Hz) (u) parameter[1]: Att. (dB) (u) parameter[2]: Phase (deg) (u)	waveform_parameter_u0 waveform_parameter_u1 waveform_parameter_u2
Code= 5	Style: Sine A/B phased Parameter number: 3	parameter[0]: DIV A (Hex) (u) parameter[1]: DIV B (Hex) (u) parameter[2]: Ini Phase (Deg) (Dec) (d)	waveform_parameter_u0 waveform_parameter_u1 waveform_parameter_d0
Code= 21	Style: Chirp Phase Coherent Parameter number: 4	parameter[0]: Fstart (ck) (Dec) (d) parameter[1]: Fstop (ck) (Dec) (d) parameter[2]: T1 (n) (Int) (u) parameter[3]: T2 (n) (Int) (u)	waveform_parameter_d0 waveform_parameter_d1 waveform_parameter_u0 waveform_parameter_u1
Code= 22	Style: Chirp Phase Continuous Parameter number: 4	parameter[0]: Fstart (ck) (Dec) (d) parameter[1]: Fstop (ck) (Dec) (d) parameter[2]: T1 (n) (Int) (u) parameter[3]: T2 (n) (Int) (u)	waveform_parameter_d0 waveform_parameter_d1 waveform_parameter_u0 waveform_parameter_u1
Code= 31	Style: RAMP Parameter number: 2	parameter[0]: DIV A (Hex) (u) parameter[1]: DIV B (Hex) (u)	waveform_parameter_u0 waveform_parameter_u1
Code= 32	Style: SINC Parameter number: 3	parameter[0]: Amp (Dec) (d) parameter[1]: Offset (Dec) (d) parameter[2]: X Scaling Factor (Dec) (d)	waveform_parameter_d0 waveform_parameter_d1 waveform_parameter_d2
Code= 50	Style: Pulse Parameter number: 5	parameter[0]: Tini (samples) (Int) (u) parameter[1]: Tr (samples) (Int) (u) parameter[2]: Tw (samples) (Int) (u) parameter[3]: Tf (samples) (Int) (u) parameter[4]: Amp (full scale) (Dec) (d)	waveform_parameter_u0 waveform_parameter_u1 waveform_parameter_u2 waveform_parameter_u3 waveform_parameter_d0
Code=100	Style: PNS Parameter number: 6	parameter[0]: Order (Int) (u) parameter[1]: Seed (Hex) (u) parameter[2]: Tr, Fraction of To (Dec) (d) parameter[3]: Tf, Fraction of To (Dec) (d) parameter[4]: To (samples) (Int) (u) parameter[5]: Amp (full scale) (Dec) (d)	waveform_parameter_u0 waveform_parameter_u1 waveform_parameter_d0 waveform_parameter_d1 waveform_parameter_u2 waveform_parameter_d2
Code=101	Style: Constant Parameter number: 1	parameter[0]: Value (Hex) (u)	waveform_parameter_u0