Creating Arbitrary Waves with DG4000

*Date:*06-11-12

This note describes the programming used in an EXCEL VBA program that loads the DG4000 with an arbitrary output. With an on-board volatile memory that can store waveforms with as many as 16K (16,384) points allows users to generate long arbitrary waveforms.

Once the waveform has been loaded into the Volatile memory, it can be saved in the Unit. To Store the waveform:

Press the Burst Button to move the Generator into Local Mode
the Remote Icon at the top of the screen should go out
Press the Store Button
Press the File Type Button then choose ARB File
Scroll To the ARB file you want to change
Press the Save Button Again
Use the Rotary switch to define the characters of the file name
the select key will enter the character/number into the name box
Press the Save Button to store the file in the DG4000

Before the example can be run on a computer the computer needs to have VISA installed. Information on getting VISA loaded on your machine can be found on the RIGOL Website. The link to the appropriate information is:
http://www.rigolna.com/search/?search=visa&x=0&y=0

RIGOL
Beyond Measure

Using the Spread Sheet

The Major Macro embedded in the spreadsheet is designed to move the waveform described by Column B into the DG4000. The graph/chart is included to give a visual representation of the data. You'll also notice that as different data is sent to the DG, the display on ARB's channel 1 changes to reflect the new pattern/waveform.

The blue background section of the SpreadSheet is used to change parameters of the data in Column B – Amplitude, Period of the Waveform, and which rows to transfer to the DG.

The check mark on the interpolation box instructs the DG to either join adjacent points with a straight line or let the output points move to discrete points – in a type of stairstep fashion.

Notes on the program

As the example is written with VBA it is easily "translated" to other languages or platforms. Reviewing the source code, modifying it, or customizing it is easily done from the DESIGN mode of EXCEL.

This example uses VISA as a structure (referenced the as RigolvisaA) that can be given a variety of parameters including

timeout

RigolvisaA.IO.Timeout = 3000

Buffer type and Length

RigolvisaA.SetBufferSize IO_OUT_BUFF, buff_size

Like most programming objects VISA has methods to take various actions. This program uses the

FindRsrc("?*") Method

to poll all the buses and return a descriptive list of all the devices connected to the computer.

instruments_found = ioMgr.FindRsrc("?")*

Using this method to find and record only resource descriptions that include the string "DG4" sorts the descriptive list and "saves" the instruments several character long descriptor - a typical descriptor for a DG4000 family looks like

USB0::6833::1601::DG4C141100138::0::INSTR

A review of the source code for this program reveals that controlling the DG4000 is accomplished by passing a number of "ENGLISH" statements to the box.

Such as using the VISA Object directly

RigolVisaA.IO.WriteString("OUT ON")

or by passing a command and an object to a function

cmd = "OUT ON"
Call writeVisa(RigolVisaA,cmd,delaytime)

In addition to these simple types of commands there is a specific command for the DG4000 family that uses a single string to define the entire waveform – even if the wave uses the entire 16384 (numbered from 1 to 16384) sample long volatile memory. The syntax for the string is

:DATA VOLATILE, Value, Value

where Value can be any value between -1.0 and + 1.0. Be aware the actual value 1.0 seems to create problem.

The actual routine used pulls data from store_array, limits the size of sample with the Format Statement, appends all the data together, and writes the string to the instrument. Subtle details of this code segment include

1) all the data values are between -1 and +1
2) the Fomat string limits the final length of the string passed to the WriteVisa subroutine
3) the IO_OUT_BUFF assigned to the RigolVisaA object that's passed to the WriteVisa function has been set to 250000 to hold the data_string that carries the output wave.

the routine that generates the string that contains the entire sample set is reproduced here.

```
 c = ":DATA VOLATILE"
data_string = ""
 first_graph_row = first_row
Last_graph_row = last_row
For i = first_row To last_row
          D = store_array(i, 1)
         D = Format(D, "0.0000")
         data_string = data_string & "," & D
Next
cmd = c & data_string
Call WriteVisa(RigolVisaA, cmd, delaytime)
```