Application Note AN6001

AnaPico IQ Data Format and Device Upload

Purpose

This application note explains the file format that has to be used to store IQ data and markers for AnaPico's Vector Signal Generators and how IQ data can be uploaded to the APVSG device using SCPI commands.

The documentation is valid for firmware ≥ 0.4.178 .

Table of Contents

Introduction

The APVSG is an ultra-fast-switching vector-modulated signal generator with an internal I/Q modulator. Customized IQ data can be loaded into internal memory and selected segments of the memory can be played back to modulate the output signal.
Marker bits can be used to wait for trigger events or to generate trigger signals (on the MF output ports).
If multiple waveforms are stored in the memory as multiple segments, it is not possible to mix segments with and without maker bits.

The application note explains the file format that must be used to store IQ values and markers. The files can be processed by the AnaPico APVSG graphical user interface to upload the data to an APVSG device. The file format is also used by the AnaPico APVSG graphical user interface to export synthesized data.

The second part of the document shows how IQ values and markers can be uploaded to the APVSG device using SCPI commands with an example written in python.

AnaPico's QI Data File Format

The "QI Data File Format" is used to generate data that can be imported by the APVSG GUI or that are exported by the GUI. The data format consists of two files, the QI Data File *.qid* and the file with meta data *.qim*.

IQ Data File: *.qid*

The data file contains the IQ samples in binary format. Each I and Q binary data point is two 16 bits two's complement values representing fixed point numbers from -1 to +1. Each sample starts with the Q data point followed by the I data point.

An **optional** byte is added to the start of the sample, containing 8 markers. Each bit represents one marker. 1: marker is on, 0: marker is off (see Figure 1)

Figure 1 File format of IQ data file

IQ Meta File: *.qim*

The meta file contains meta data about the data in the IQ data file. The following tags are supported. Other tags and comments are ignored. If *.qim* file is missing, default parameters are assumed (no marker byte).

Example meta file

```
# comments are ignored
version = 1.0
dataFile = exampleFile.qid
description = any string to describe the modulation
dateCreated = 2021-04-28-13:20:58
numberOfSamples = 10000
markerBits = 8
```

IQ Sequence File: *.qis*

Each sequence file contains a script which describes a single sequence. For information on the concept and rules for sequence scripts, please consult AN6003.

IQ File: *.qi* (for legacy reasons)

For legacy reasons a file with file extension *.qi* can be used that is limited to only the IQ data points in binary format without marker.

Upload IQ data to APVSG by SCPI

IQ modulation samples can be sent to the vector signal generator with a SCPI block transfer.
Please note:

- Writing waveforms does not overwrite waveforms existing on the device.
- Waveforms shorter than the minimum number of samples required per waveform will be automatically extended by cyclically repeating the waveform [2].
- The remaining number of samples that may be written to the device can be queried using *BB:ARBitrary:WAVeform:DATA:FREE?*
- To clear the content of the device memory use the *BB:ARBitrary:WAVeform:DATA:DELete ALL* command.
- Before sending any waveform, the device must be configured to handle IQ data with or without markers.
  When enabled, each IQ sample features additional marker bits that can be set individually. Using markers increases the logical size of a sample and thus reduces the total number of samples that can be stored on the device.

For more information, please refer to the *APVSG Data Sheet* [2]

This command enables or disables marker bits

BB:ARBitrary: WAVeform:MARKer:STATe ON|OFF|1|0

The following command writes waveforms (IQ modulation data samples with optional marker bits, but no meta information) to the device.

BB:ARBitrary:WAVeform:DATA [<integer>],<data>

Parameter 1, [<integer>]

This optional parameter specifies the segment index used to store more than one waveform. For more information, please refer to [3].

Parameter 2, <data>

Data sent or received has IEEE488.2 definite block data format:

#<num_digits><byte_count><data byte>{<data_byte}

<num_digits> specifies how many digits are contained in <byte_count>.

<byte_count> specifies how many data bytes follow in <data_bytes>.

Example of definite block data:

#18xxxxxxxx

#18…: byte count is one digit wide

#1**8**…: 8 data bytes will follow

…**xxxxxxxx**: 8 bytes of data

The data itself consists of IQ data samples and optional marker bits. An IQ data sample is 32 bits wide (without marker bits) or 40 bits wide (with marker bits) and contains two 16 bits two's complement values representing fixed point numbers from -1 to +1.

Data format with marker bits:

Byte	Sample	Content
0	1	8 marker bits
1	1	Lower (least significant) 8 bits of Q (quadrature) component.
2	1	Higher (most significant) 8 bits of Q (quadrature) component.
3	1	Lower (least significant) 8 bits of I (in-phase) component.
4	1	Higher (most significant) 8 bits of I (in-phase) component.
5	2	8 marker bits
6	2	Lower (least significant) 8 bits of Q (quadrature) component.
7…	2…	…

The clock rate matching the playback rate of IQ data:

BB:ARBitrary:CLOCk <float>

The IQ modulation based on waveforms stored in the memory can be enabled or disabled with:

BB:ARBitrary:WAVeform:STATe ON|OFF|1|0

For further information please refer to the AnaPico Signal Generator Programmers Manual [1] and the APVSG datasheet [2].

SCPI Example

Complete SCPI command sequence for modulating a 1 GHz / 0 dBm RF carrier with IQ data and marker bits from memory:

SOUR 1	Select output channel 1
OUTP ON	Enables RF output
FREQ 1e9	Sets initial RF output frequency to 1 GHz
POW 0	Sets RF output power to 0 dBm
BB:ARB:CLOC 500e6	Set baseband sampling clock to 500 MHz
BB:ARB: WAV:MARK:STAT ON	Configure the device to use markers
BB:ARB:WAV:DATA:DE ALL	Delete memory content
BB:ARB:WAV:DATA <block data>	Uploading data (marker bits and IQ samples)
***OPC?**	Check if upload completed
BB:ARB:WAV:STAT ON	Turn IQ modulation on

Further Documentation

[1] AnaPico Programmer's Manual for Signal Generators
https://www.anapico.com/downloads/manuals/

[2] AnaPico APVSG Datasheet
https://www.anapico.com/downloads/brochures-and-data-sheets/

[3] AnaPico AN6003 APVSG - Memory Segmentation
https://www.anapico.com/downloads/application-notes-and-videos/

Example Scripts

Create qid/qim files using python

```
import numpy as np
import math
import ctypes
from datetime import datetime

n = 10000 # number of IQ pairs (representing 4 bytes each)
v = 32768 # maximum amplitude of values

# create IQ values
####################

iValues = np.ones(n,);
qValues = np.ones(n,);
for i in range(n):
    iValues[i] = v * math.sin(2 * math.pi * i * 300.0 / n)
    qValues[i] = v * math.cos(2 * math.pi * i * 300.0 / n)

# save to data file
f = open('saveFile.qid', 'wb')
for i in range(n):
    f.write((int)(0b00000001).to_bytes(1, byteorder='little', signed=False)) # marker
    f.write((int)(qValues[i]).to_bytes(2, byteorder='little', signed=True)) # Q
    f.write((int)(iValues[i]).to_bytes(2, byteorder='little', signed=True)) # I
f.close()

# save to meta file
####################

f = open('saveFile.qim', 'w+')
f.write('version = 1.0\n')
f.write('dataFile = saveFile.qid\n')
f.write('dateCreated = 2021-04-28-13:20:58\n')
f.write('description = 1-tone offset\n')
f.write('sequenceID = 1\n')
f.write('numberOfSamples = 10000\n')
f.write('samplingRate = 500e6\n')
f.write('markerBits = 8\n')
f.close()
```