

## Introduction

Holzworth Instrumentation allows customer access to Instrument DLLs for use with alternate interface and control programs. The HolzworthHS1001A.dll allows direct access to the HS1001A through any software that can call a standard C DLL. Both integer and string commands are available.



## General Usage

For the HS1001A, all DLL functions require that a specific unit serial number be used. Using specific serial numbers simplifies the programming of multiple HS1001A synthesizers because each device has a unique serial number. If a Holzworth virtual instrument (HS1001) is swapped in an automated test station, only the variable representing the serial number need be replaced to operate the alternate instrument.

Note that only one modulation function or sweep function may be active any time. Therefore, the Sweep and Modulation functions may not be combined. To shut off Sweep or Modulation, use the "ModEnableNo" command.

## Commands : HolzworthHS1001A.dll

```
int openDevice(const char *serialnum)
int RFPowerOn(const char *serialnum)
int RFPowerOff(const char *serialnum)
int ModEnableNo(const char *serialnum)
int ModEnableFM(const char *serialnum)
int ModEnableAM(const char *serialnum)
int ModEnablePM(const char *serialnum)
int ModEnablePulse(const char *serialnum)
int ModEnableSweep(const char *serialnum)
int setPower(const char *serialnum, short powernum)
int setPowerS(const char *serialnum, const char *powerstr)
int setPhase(const char *serialnum, short phasenum)
int setPhaseS(const char *serialnum, const char *phasestr)
int setFrequency(const char *serialnum, INT64 frequencynum)
int setFrequencyS(const char *serialnum, const char *frequencystr)
int setFrequencyDwell(const char *serialnum, unsigned short dwellnum)
int setFrequencyDwellS(const char *serialnum, const char *dwellstr)
int setFrequencyPoints(const char *serialnum, unsigned short pointsnum)
int setFrequencyPointsS(const char *serialnum, const char *pointsstr)
```

**Command Descriptions****int openDevice(const char \*serialnum)**

- Opens the device by serial number
- Returns a 1 for success, 0 for failure

**int RFPowerOn(const char \*serialnum)**

- Turns the RF Power On
- Returns a 1 for success, 0 for failure

**int RFPowerOff(const char \*serialnum)**

- Turns the RF Power Off
- Returns a 1 for success, 0 for failure

**int ModEnableNo(const char \*serialnum)**

- Turns the All Modulation and Sweep functions off – CW mode
- Returns a 1 for success, 0 for failure

**int ModEnableFM(const char \*serialnum)**

- Enables FM modulation. If sweep was previously enabled, will revert back to CW mode before enabling FM modulation.
- Returns a 1 for success, 0 for failure

**int ModEnableAM(const char \*serialnum)**

- Enables AM modulation. If sweep was previously enabled, will revert back to CW mode before enabling AM modulation.
- Returns a 1 for success, 0 for failure

**int ModEnablePM(const char \*serialnum)**

- Enables PM modulation. If sweep was previously enabled, will revert back to CW mode before enabling PM modulation.
- Returns a 1 for success, 0 for failure

**int ModEnablePulse(const char \*serialnum)**

- Enables Pulse modulation. If sweep was previously enabled, will revert back to CW mode before enabling Pulse modulation
- Returns a 1 for success, 0 for failure

**int ModEnableSweep(const char \*serialnum)**

- Enables Sweep. If modulation was previously enabled, it will stop modulation functions
- Returns a 1 for success, 0 for failure

**int setPower(const char \*serialnum, short powernum)**

- Sets the RF output power. Send a 16bit signed integer in tenths of a dB
- Example: 10dBm, send 100, for -105.4dBm, send -1054
- Returns a 1 for success, 0 for failure

**int setPowerS(const char \*serialnum, const char \*powerstr)**

- Sets the RF output power. Send a string without the decimal to the tenths of a dB.
- Example: 10dBm, send '100', for -105.4dBm, send '-1054'
- Returns a 1 for success, 0 for failure

**int setPhase(const char \*serialnum, short phasenum)**

- Sets the RF phase. Send a 16bit signed integer in tenths of a degree
- Example: 10°, send 100, for -105.4°, send -1054
- Returns a 1 for success, 0 for failure

**int setPhaseS(const char \*serialnum, const char \*phasestr)**

- Sets the RF phase. Send a string without the decimal to the tenths of a degree
- Example: 10°, send '100', for -105.4°, send '-1054'
- Returns a 1 for success, 0 for failure

**int setFrequency(const char \*serialnum, INT64 frequencynum)**

- Sets the RF Frequency. Send a 64bit unsigned integer in milliHertz (mHz)
- Example: 16.1MHz, send 16100000000
- Returns a 1 for success, 0 for failure

**Caution: Some software (Matlab) has problems with 64bit integers and dlls. Use the string version if this is an issue.**

**int setFrequencyS(const char \*serialnum, const char \*frequencystr)**

- Sets the RF Frequency. Send a string in milliHertz (mHz)
- Example: 16.1MHz, send '16100000000'
- Returns a 1 for success, 0 for failure

**int setFrequencyDwell(const char \*serialnum, unsigned short dwellnum)**

- Sets the Sweep Dwell Time. Send a 16 bit integer in milliseconds (ms)
- Example: 26ms, send 26
- Returns a 1 for success, 0 for failure

**int setFrequencyDwellS(const char \*serialnum, const char \*dwellstr)**

- Sets the Sweep Dwell Time. Send a string in milliseconds (ms)
- Example: 26ms, send '26'
- Returns a 1 for success, 0 for failure

**int setFrequencyPoints(const char \*serialnum, unsigned short pointsnum)**

- Sets the Sweep Number of Points. Send a 16 bit integer with number of points
- Example: 1001 points, send 1001
- Returns a 1 for success, 0 for failure

**int setFrequencyPointsS(const char \*serialnum, const char \*pointsstr)**

- Sets the Sweep Number of Points. Send a string with number of points
- Example: 1001 points, send '1001'
- Returns a 1 for success, 0 for failure

For further details, refer to Holzworth Application Notes:

**AN101-1.0      USB HID Software Architecture**  
**AN103-1.0      Using the HolzworthHS1001A.dll with Matlab™**

Contact Holzworth Instrumentation by phone or email for technical support:

**Phone:    +1.303.235.3473**  
**Email:    [support@holzworth.com](mailto:support@holzworth.com)**