NANO-DRIVE® OPERATION MANUAL

MAD CITY LABS INC.

2524 Todd Drive, Madison, WI 53713
Tel: (608) 298-0855 Fax: (608) 298-9525
www.madcitylabs.com sales@madcitylabs.com

IMPORTANT SAFETY INFORMATION

Hazardous voltages and currents present in controller and stage. Risk of electrocution and death.
Do not open controller enclosure or stage body. There are no user serviceable parts.

Piezoactuators have large capacitance and are capable of storing hazardous amounts of electrical energy over long periods of time. Various conditions such as load and temperature changes can also cause piezoactuators to accumulate charge.

It is recommended that the input command voltage be set to zero and turn off the Nano-Drive® before disconnecting the nanopositioner from the controller.

The Nano-Drive® has no user serviceable parts. Only trained service personnel should perform service. Contact: sales@madcitylabs.com or +1 608 2980855 (8:00am to 5:00pm US Central standard time). Unauthorized service will void the warranty.

IMPORTANT

All Technical Information, recommendations, and examples related to Mad City Labs, Inc. products made in this manual are based on information believed to be correct. The purchaser or user should determine the suitability of each product before using. The purchaser or user assumes all risks and liability whatsoever in connection with the use of any and all Mad City Labs, Inc. products or services.

CONTENTS

1 INTRODUCTION

The Nano-Drive® is the complete control package required for operating all Mad City Labs, Inc. nanopositioning systems. The Nano-Drive® controller and matching nanopositioning stage are sensitive scientific equipment designed for nanometer scale positioning. It is recommended for optimum performance that both controller and stage be operated in low electrical and acoustic noise environments.

The Nano-Drive® includes a low noise, low drift 150 V high voltage driver, PicoQ® position sensing electronics, and closed loop proportional-integral feedback control. The driver is capable of supplying 50 mA at 150 V. The output current is short circuit protected. The closed loop position command may be supplied through the front panel analog interface or, alternatively, by the Universal Serial Bus (USB) digital interface (if option is installed).

The Nano-Drive® is available in one, two and three axis versions. Each Nano-Drive® has been factory adjusted for its complementary nanopositioning device. This adjustment includes setting the parameters of the position sensing electronics. Under stable environmental conditions readjustment of these parameters is not necessary. Each nanopositioning stage and each axis on the Nano-Drive® are clearly labeled for identification.

Hazardous voltages and currents present in controller and stage.
Risk of electrocution and death.
Do not open controller enclosure or stage body. There are no user serviceable parts.

Piezoactuators have large capacitance and are capable of storing hazardous amounts of electrical energy over long periods of time. Various conditions such as load and temperature changes can also cause piezoactuators to accumulate charge.

OUTPUT CURRENT (Maximum)	50 mA
ANALOG INTERFACE * (INPUT BNC) (Standard) (AR-5 Option) (AR-6 Option) (AR-10 Option)	 0.0 to +10.0V -5.0 to +5.0V -6.0 to +6.0V) -10.0 to +10.0V
FRONT PANEL ACCESSIBLE OUTPUT SIGNALS Output Voltage ÷ 10 (HV/10) Position Sensor Signal	 -2.0 to +15.0V 0.0 to +10.0 V
FRONT PANEL ADJUSTMENTS (Standard) (SO Option) (OCL Option)	 SENSOR OFFSET SCAN OFFSET OPEN/CLOSED LOOP
CONNECTOR TYPE	DB-9
POWER REQUIREMENT	12VDC/5.17A
DIGITAL INTERFACE *(OPTION)	USB 16 OR 20 BIT
OTHER ACCESSIBLE SIGNALS (ISS Option)	 TTL (4)

Table 1: Nano-Drive® specifications

*The command signal is the sum of the digital and analog interface inputs.

A block diagram of the Nano-Drive® is shown in Fig.1. The command signal is used to control the displacement of the nanopositioning device. This command signal can be supplied by either the front panel BNC analog input or by the internal DAC of the USB interface. A command signal of minimum to maximum voltage values corresponds to a stage displacement of zero to the maximum value. If the USB interface is to be used alone, it is recommended that the front panel BNC be grounded.

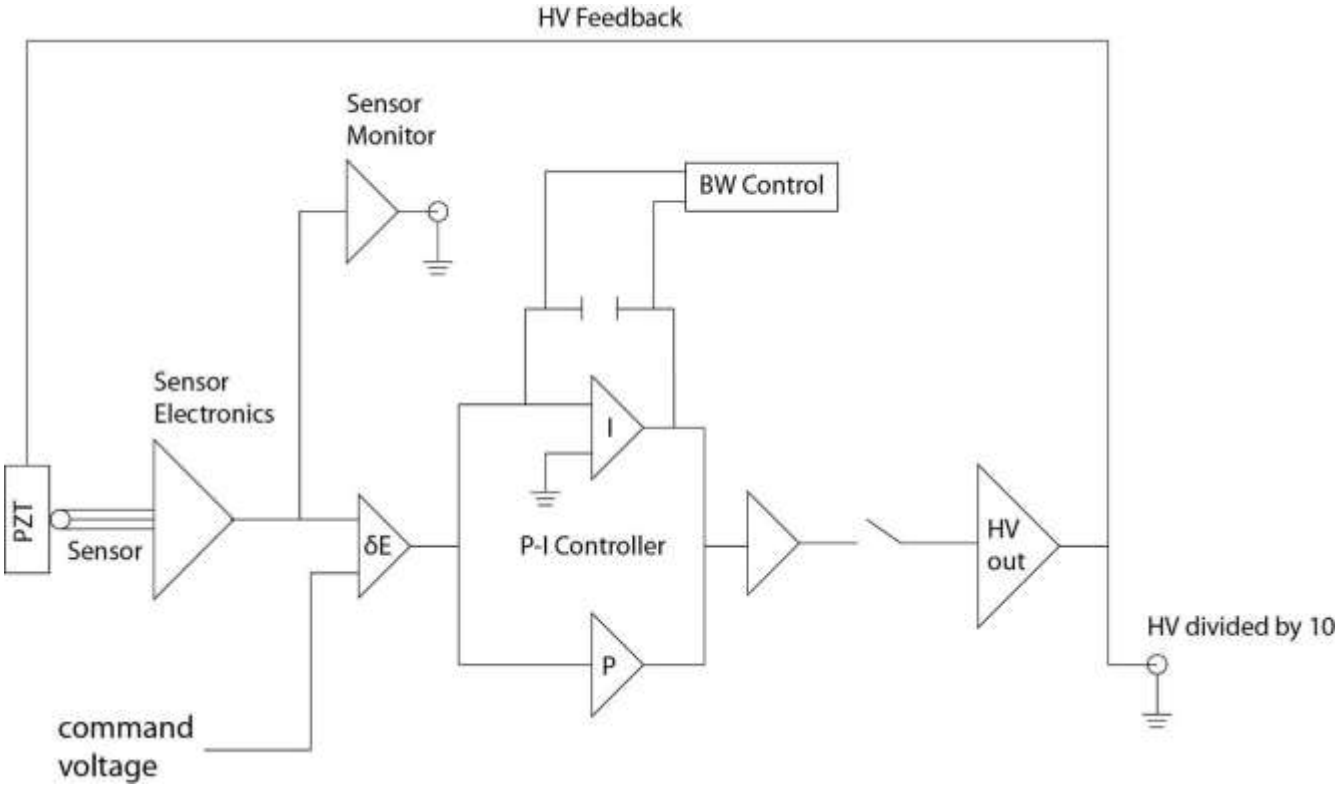


Figure 1: Block diagram of Nano-Drive®

The command signal (cmd) is compared to the sensor signal and an error signal δE is generated. The error signal is passed on to the proportional-integral (PI) controller. The stage displacement is held at the value corresponding to the command signal, eliminating the effects of creep and hysteresis.

1.1 Front panel connections

Figure 2: Nano-Drive® Controller Front Panel

The front panel of a three axis Nano-Drive® controller is shown in Fig.2. Each axis has the following front panel connections

Input	Analog input for command voltage (See Table 1 for values)
Sensor output	0.0 to 10.0 V buffered output on a BNC connector
DB-9	9-Pin D-Type connector to nanopositioning stage (1 per axis)
Offset	adjustment for position sensing circuit
HV/10	HV÷10 (-2.0V to +15.0V) output on a BNC connector

1.2 Rear panel connections

Figure 3: Nano-Drive® Controller Rear Panel

The rear panel connections for the Nano-Drive® controller are shown in Fig.3.

DC-Power	12V/5.17A
USB[1] port	USB Connector (USB port for single axis on front panel)
Pixel Clock Output[2]	250ns TTL compatible pulse synchronized with the X-axis ADC
Line Clock Output[2]	250ns TTL compatible pulse (user control)
Frame Clock Output[2]	250ns TTL compatible pulse (user control)
Aux Clock Output[2]	250ns TTL compatible pulse (user control)

[1]USB option only

[2]ISS option only

The DC power cable connector locks to the enclosure connector. To remove the power cable, grasp the rectangular housing of the cable and pull away from the enclosure connector. Do not pull the cable directly as this will not release the locking mechanism.

1.3 Available Options

Table 2 describes the available options for the Nano-Drive® controller.

Name	*Description*	*Remarks*
USB16x	16 bit USB interface for (x=1…3) axes	Not compatible AR-5, AR-6, AR-10 options
USB20x	20 bit USB interface for (x=1…3) axes	Compatible with all input voltage ranges
ISS	Integrated Scan Synchronization	Only available with USB options
AR-5	Analog input voltage range (-5.0 to +5.0V)	Compatible with USB20x only
AR-6	Analog input voltage range (-6.0 to +6.0V)	Compatible with USB20x only
AR-10	Analog input voltage range (-10.0 to +10.0V)	Compatible with USB20x only
OCL	Open/Closed loop	Front panel switch to select between open and closed loop modes (see section 1.3.2)
SO	Scan offset	Not compatible with AR-5, AR-6, AR-10 options. (see section 1.3.3)
RM	Rack mounting hardware	
VBOB	Veeco Breakout Box	Please contact our engineers about the suitability of this option for your instrumentation.
ND45x	Upgrade from standard controller to 45W output power for (x=1…3) axes	Upgrade applicable to entire controller not just selected axes. Contact our engineers about this option.
ND85x	Upgrade from standard controller to 85W output power for (x=1…3) axes	Upgrade applicable to entire controller not just selected axes. Contact our engineers about this option.

Table 2: Available options

1.3.1 AR- Options

Controllers enabled with the AR-type options have input command voltages that range from a minimum (negative) voltage value to a maximum (positive) voltage value. The number after the AR- designation describes the voltage value (see table 2 above). The AR-type option is compatible with analog interface only Nano-Drive® controllers and 20 bit USB enabled Nano-Drive® controllers.

Users with the analog interface only controllers should note that if no external voltage is applied to the analog input BNC, the stage will immediately move to the midpoint of displacement upon powering up the Nano-Drive®. This does not cause any damage to the stage.
Users with USB enabled controllers should note that the commands from the USB and the analog input are summed together. The default position of the USB input is zero displacement at start up. Zero displacement corresponds to the negative minimum voltage (e.g. -10V for AR-10 option). It is important to realize that this negative voltage will be summed to any voltages applied to the analog input and that users should ensure that they understand the relationship between voltages and displacement of their stage. Examples are discussed in section 3.2.

1.3.2 OCL (Open/Closed Loop) Option

Controllers enabled with the OCL option are able to drive the nanopositioning stage in open loop or closed loop mode. The power to the Nano-Drive® must be turned off before switching modes. If the power is not switched off a transient jump in position will occur as the feedback loop is connected or disconnected and will cause damage to the piezoactuators. The nanopositioner in open loop mode can be driven at higher frequencies that may approach the resonant frequency of the stage. This will result in instability or oscillations of the stage and should be avoided. If you have any questions about running this system in open loop mode please call our technical support personnel.

1.3.3 SO (Scan Offset) Option

For controllers enabled with the SO option, the front panel scan offset potentiometers add a voltage to the input command voltage. The value added is zero when the potentiometer is rotated fully counterclockwise and 10.0V when the potentiometer is rotated fully clockwise. Since all command voltages (analog input voltage, USB commanded voltage, scan offset) are summed it is important to ensure that the scan offset potentiometer is set to zero when not in use. Do not allow the summed voltage values to exceed the maximum input voltage range.

1.3.4 VBOB Option

Controllers supplied with the VBOB option allow the user to interface between the *Veeco Signal Access Module (SAM)* and the Nano-Drive® controller. The VBOB enclosure has been configured to accept two ±220V signals and one ±12V signal (nominally X,Y and LV Z (SPM)

output signals from the SAM). The VBOB enclosure then outputs three ±10V signals which can then be inputted safely to the Nano-Drive® controller equipped with the AR-10 option.
Supplied with the VBOB enclosure are SHV BNC cables to connect the SAM outputs to the VBOB inputs (±220V) and standard BNC cables to connect the ±10V output from VBOB to the input of the Nano-Drive® controller.

1.3.5 Nano-Drive® power upgrade option

For some applications, such as high speed scanning, it may be necessary or desirable to draw on more power from the Nano-Drive® controller. There are two available options for this upgrade: 45W and 85W power upgrades. These options are applicable to the entire controller and not just a selected axis. If you have this option enabled it will be indicated on the front panel.

1.4 Recommended Operating Practice

Irrespective of what configuration of the Nano-Drive® controller you own, we recommend that you establish a known default setting for your system. For most users we recommend setting the analog input voltages to 0V prior to turning off the Nano-Drive® controller or during extended periods of inactivity. For the majority of users this will also correspond to a zero stage displacement (exception being those with AR-type options installed).
The internal DAC of the USB interface defaults to the zero displacement value at start up, position values are not retained once power is switched off to the Nano-Drive® controller. It is recommended that users establish a default setting that is relevant to their needs, particularly those with USB enabled, AR-type controllers.

2 INSTALLATION

Prior to commencing installation of your Nano-Drive® controller, be certain to install your nanopositioning stage as described in its **"Installation and Operation Manual"**. Each DB-9 connector from the stage is clearly identified for a specific axis. It is important that each stage axis is connected to the corresponding Nano-Drive® axis.

2.1 Initial Installation for Controllers with Analog Interface Only

a) Be certain the Nano-Drive® power switch is off.

b) Connect the DB-9 connector(s) of the nanopositioning stage to the Nano-Drive® and tighten down the screws.

c) Set the Analog Command (BNC) signal voltage to the minimum voltage. This value should correspond to zero displacement of the stage.

d) Turn on the Nano-Drive®.

e) The command signal will now control the displacement of the nanopositioning stage.

f) Check the sensor offset value. Section 2.3 describes the normal operating value and procedure for adjustment.

g) The system is now ready for operation (see section 3)

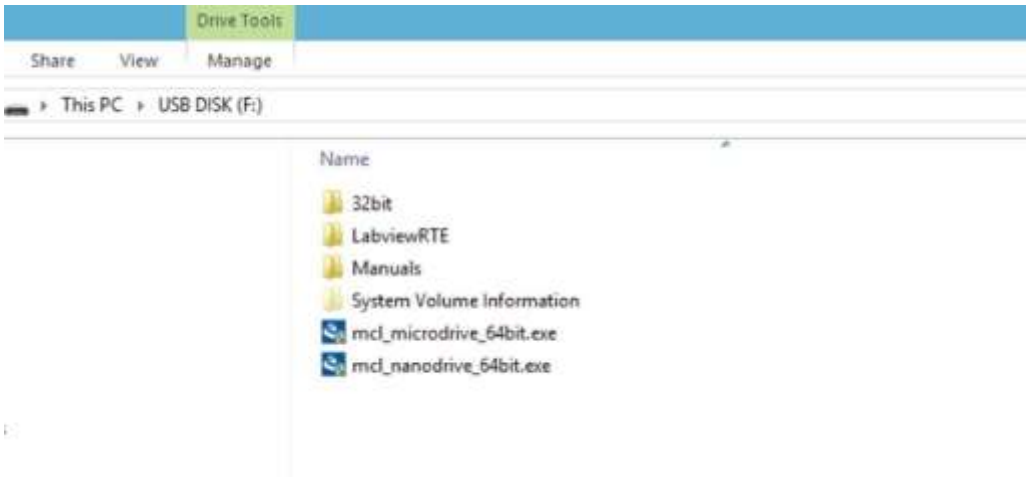
h) Always set the analog command signal to 0.0V (or your default setting) before turning off the system.

i) Never disconnect the DB-9 connector(s) while the power is on. Wait one minute to allow the internal piezoactuators to discharge before removing the DB-9 connector.

2.2 Initial Installation for USB Enabled Controllers

The USB driver and software must be installed on your host computer before connecting the Nano-Drive® controller to the host computer. Your host computer must be using either Windows Vista, 7,8, or 10.

The installation program is on the Manuals and Software USB flash drive shipped with your system. The installation program does **NOT** automatically run when the USB flash drive is inserted into the port. The USB flash drive contains software installations suitable for 64 bit operating systems and 32 bit operating systems, copies of the relevant manuals, and a copy of the LabView runtime engine (RTE) for users planning on using executable versions of our LabView based examples and software.



Initial screen after inserting the USB Flash Drive

The installations for 64 bit operating systems are located in the root directory. If you are using 32 bit software on a 64 bit operating system you must use the 64 bit installation. In the screenshot above "**mcl_nanodrive_64bit.exe**" is the installation file to use if you have are using a Nano-Drive® controller on a 64 bit operating system.
If you are running older, 32 bit operating systems the software installation executable is located in the 32 bit folder.

The installation program installs the example software and manuals for your system. The installed information is at C:\Program Files\Mad City Labs.
Users planning on using the LabView executable programs should manually install the LabView runtime engine located in the LabView RTE folder on the USB flash drive.

Once the software has been installed, connect the PC to the controller. Windows will now recognize that a new USB device has been attached to your computer and install the required software. No further steps are required.

With the software and nanopositioning stage installed the following steps can be performed.

a) Be certain the Nano-Drive® power is disconnected on the rear panel.

b) Connect the DB-9 connector of the nanopositioning stage to the Nano-Drive® and tighten down the screws.

c) Ensure that there is no voltage to the analog command (BNC). The simplest approach is to leave it disconnected from any external voltage source. The USB internal DAC will produce the only available voltage command (default is zero displacement).

d) Connect power supply to the rear panel of the Nano-Drive®.

e) The USB “ready” light should come on indicating that the USB firmware has been loaded from the onboard EEPROM into the onboard microcontroller.

f) If using the USB interface, the Nano-Drive® may now be connected to your host computer with the provided USB cable (standard USB cables may also be used).

g) Check the sensor offset value. Section 2.3 describes the normal operating value and procedure for adjustment.

h) The system is now ready for operation.

i) Never disconnect the DB-9 connector(s) while the power is on.

2.3 Sensor Offset Adjustment

******IMPORTANT******

Before commencing data collection or measurement it is important to check the sensor offset value. We recommend that this be an integral part of your initialization procedure prior to data collection and whenever circumstances have changed in your instrumentation arrangement (e.g. temperature changes, fixture changes etc.).

******IMPORTANT******

The offset of the position sensing electronics have been factory adjusted. In multiple axis systems, it is important to connect the correct nanopositioning stage to each axis on the Nano-Drive®. These axes are clearly marked on both the Nano-Drive® and the nanopositioning stages.

In closed loop mode the high voltage output operates within a specific voltage range. The high voltage output incorporates extra voltage at either end of the range to facilitate the closed loop feedback. Once the new nanopositioning stage is installed it is advisable to adjust the offset of the position sensing circuit. Once adjusted, no further adjustment should be necessary unless the stage is reinstalled or the temperature changes by more than 5°C.

IT IS ALWAYS BEST TO CONTROL THE ENVIRONMENTAL CONDITIONS RATHER THAN CONTINUOUSLY ADJUSTING THE SENSING ELECTRONICS. USE THE NANOPOSITIONING SYSTEM IN A TEMPERATURE CONTROLLED ROOM AND MINIMIZE THE NUMBER OF TIMES THE STAGES ARE REINSTALLED.

2.3.1 Sensor offset adjustment for controllers with analog interface only

Use the following procedure to set/adjust the offset:

a) With the nanopositioning stage installed and connected to the Nano-Drive® and the Nano-Drive® on, monitor the HV/10 output signal using a calibrated DVM.

b) Apply the minimum voltage (e.g. 0.0V for standard interface, -10.0V for AR-10 option etc.) to the command signal input. The stage should be at the zero displacement.

c) If the HV/10 value is between -1.2V and -1.7V then no further adjustment is necessary. If the HV/10 value is not in the desired range then proceed to step (d). *Please note that HV/10 can be up to 15.0V.*

d) Using an insulated screw driver, adjust the front panel offset screw so that the HV/10 output signal is between -1.2V and -1.7V. Adjust very slowly while monitoring the signal. Clockwise increases the voltage.

2.3.2 Sensor offset adjustment for USB enabled controllers

Use the following procedure to set/adjust the offset:

a) With the nanopositioning stage installed and connected to the Nano-Drive® and the Nano-Drive® on, monitor the HV/10 output signal using a calibrated DVM.

b) Ensure that there is no voltage to the analog command (BNC). The simplest approach is to leave it disconnected from any external voltage source. The USB internal DAC will produce the only available voltage command (default is zero displacement).

c) If the HV/10 value is between -1.2V and -1.7V then no further adjustment is necessary. If the HV/10 value is not in the desired range then proceed to step (d). *Please note that HV/10 can be up to 15.0V.*

d) Using an insulated screw driver, adjust the front panel offset screw so that the HV/10 output signal is between -1.2V and -1.7V. Adjust very slowly while monitoring the signal. Clockwise increases the voltage.

3 OPERATING THE NANO-DRIVE® CONTROLLER VIA ANALOG COMMANDS

The standard Nano-Drive® controller provides an analog interface via front panel mounted BNC connectors. Each BNC requires an input voltage to control the displacement of the connected nanopositioning stage. The input signal is typically supplied via a stable power supply or via a digital to analog converter (DAC) on a data acquisition card.

The standard input range is 0.0V to 10.00V corresponding to a zero to maximum stage displacement. For systems ordered with AR-type options or other custom input ranges, the minimum input voltage corresponds to the zero stage displacement and the maximum input voltage corresponds to the maximum stage displacement. Irrespective of which input range is installed, do not exceed the maximum value on the analog input and do not go below the minimum voltage.

USB enabled Nano-Drive® controllers can be operated via analog commands. Section 3.1 describes operating a USB enabled Nano-Drive® with the standard analog input voltage range. Section 3.2 describes operating a USB enabled Nano-Drive® that is equipped with an AR-type option. It is important if your controller is USB enabled that you read the relevant section before operating your system via analog commands.

When not in use it is advisable to leave the Nano-Drive® on with the command signal voltage at 0V or your chosen default value. Never disconnect the DB-9 connector when the Nano-Drive® is on.

3.1 Operating the USB enabled Nano-Drive® via analog commands (0 to 10V input)

Nano-Drive® controllers with either USB option can be operated via (1) analog interface only, (2) USB digital interface only or, (3) analog and digital commands.
Operating the Nano-Drive® via the USB interface only is covered in sections 4 (USB 16 bit) and 5 (USB 20 bit). Irrespective of which mode of operation you choose it is important to note that the analog input command voltage is summed with the digital command voltage supplied via the USB interface.

The analog inputs, when not connected to an external voltage source, are held at 0V. The USB interface automatically defaults to the zero position at startup. For controllers with the standard

input range (0 to 10V), zero position corresponds to a 0V input from the USB interface. Therefore, for standard controllers the default start up position is the zero position for the stage.

When not in use it is advisable to leave the command signal voltage at the minimum (0V) and the Nano-Drive® on. Never disconnect the DB-9 connector when the Nano-Drive® is on.

3.2 Operating USB enabled Nano-Drive® via analog commands (AR- options)

USB enabled Nano-Drive® controllers installed with an AR- option (AR-5, AR-6, AR-10) can be operated via (1) analog interface only, (2) USB digital interface only or, (3) analog and digital commands. Operating the Nano-Drive® via the USB interface only is covered in sections 4 and 5. Irrespective of which mode of operation you choose it is important to note that the analog input command voltage is summed with the digital command voltage supplied via the USB interface.

The analog inputs, when not connected to an external voltage source, are held at 0V. The USB interface automatically defaults to the zero position at startup. For controllers with AR- options installed, zero position corresponds to the minimum voltage input from the USB interface. The minimum voltage value is always a negative value (e.g. -10V for AR-10 option).

Analog input (V) + digital input (V) = commanded voltage value (V) = position (microns)

The examples below show the consequences of adding the command voltages.

Example 1: Initial startup (AR-10 option)

0 + (-10)V = -10V = 0 microns

The stage is in the zero position at startup

Example 2: Commanding via the analog input (AR-10 option)

The user wishes to command via the analog input only. Since the AR-10 option is installed the range for the analog input range is -10V to +10V. The user would like to begin at the zero position of the stage therefore a command of -10V is placed via the analog input.

(-10) + (-10)V = -20V = ?

Clearly this is not the desired result! The command voltages sum to produce a voltage that does not correspond to a position (since the range only extends to -10V). The solution is to ensure

that the digital command is set to zero volts, which corresponds to the midpoint of the displacement range.

When the digital command is set to the midpoint of displacement (0V) then;

(-10) + (0)V = -10V = 0 microns

This is the desired result. **Therefore, if the user intends to command the stage via analog control only, the digital command should be always set to the midpoint of displacement (0V).** The stage now runs from the zero position to the maximum position based on the analog input commands.

The digital position command remains in memory provided the Nano-Drive® power is on. When not in use it is advisable to leave the command signal voltages at a chosen default value suited to your operation and the Nano-Drive® on. Never disconnect the DB-9 connector when the Nano-Drive® is on.

4 OPERATING THE NANO-DRIVE® VIA THE USB 16 BIT INTERFACE (OPTION)

This section describes the operation of the Nano-Drive controller via the USB 16 bit interface option operating under Microsoft Windows. The optional USB interface in the Nano-Drive® controller allows the user to digitally control the position of up to three axes. The position of each axis is controlled independently by a dedicated DAC and read independently by a dedicated ADC. Rate limitations for single read/write actionson the ADC/DAC are listed in Table 3 below. For information about implementation of read/write actions see sections 4.2-4.4. Communication with the host PC is controlled by the installed USB driver, and the driver is accessed by function calls to the MADLib.dll library. The position, in micrometers, of the nanopositioner can be changed and monitored under software control by using the supplied dynamically linked library (.dll).

Table 3: ADC/DAC rate limitations for single read/write actions

Product	***ADC & DAC rates for single read/write***
1-Axis	5ms/point to 33µs/point (in 83.33ns steps) **(33µs/point is the default rate)**
2-Axis	5ms/point to 100µs/point (in 83.33ns steps) **(100µs/point is the default rate)**
3-Axis	5ms/point to 100µs/point (in 83.33ns steps) **(100µs/point is the default rate)**

Faster rates may apply under waveform acquisition. See the madlib_*_*.doc for more information.

4.1 Introduction to MADLib.dll

Windows allows programs to link to function calls in either static or dynamic libraries. For a program to link to a static library, it must be incorporated when the program is compiled, while for a dynamically linked library, the program is linked to the library when it is run. Programs written in LabView or Visual Studio (C++, C#, VB etc.) can make function calls to dynamically linked libraries such as MADLib.dll. In LabView, these function calls to an external DLL are referred to as “call library functions”. The user may wish to study the many LabView examples

provided and read the LabView documentation to get a better understanding of how to used “call library functions”. In order to view the LabView examples you must have LabView installed (not supplied).

The USB driver provided with the Nano-Drive® is designed to mediate communications between Windows and the Nano-Drive® . Communication between the USB driver and the user is controlled by MADLib.dll When a user wishes to access a USB device such as the Nano-Drive® , a “handle” to the device must be opened in Windows as a means for the user and Windows to identify a specific device the user wishes to communicate with. In this sense, a “handle” is simply a unique identifier.

All programs should be of the form:

<...>

Initialize handle(s).

Program portion that relies on the Nano-Drive®(s).

Release handle(s).

<...>

4.1.1 Using a single Nano-Drive® controller

Using a single controller three functions are required to properly manage handles:

- MCL_InitHandle will obtain control of the controller. Should a LabView program stop/crash/finish prior to releasing the handle subsequent calls to MCL_InitHandle will fail since the DLL still has the handle.
- MCL_InitHandleOrGetExisting will obtain control of the controller. This function will return control of the controller even if the LabView program did not properly release the handle.
- MCL_ReleaseHandle will relinquish control of the controller.

4.1.2 Using multiple Nano-Drive® controllers

Using multiple Nano-Drives MCL_InitHandle and MCL_GrabHandle may have ambiguous results. MCL_InitHandle will obtain a handle to one of the Nano-Drive®(s) attached. MCL_GrabHandle may be used to differentiate between different classes of Nano-Drive® (s), however, if multiple devices of the same class are attached the result will be ambiguous. To deal with this ambiguity each Nano-Drive® has one unique feature, namely, its serial number. MCL_GrabAllHandles may be used to obtain handles to all Nano-Drive®(s) attached but not yet under control. MCL_GetHandleBySerial may be used to search all of the attached and controlled Nano-Drive®(s) for a Nano-Drive® with a matching serial number.

More detailed documentation on these functions can be found in the document Madlib_*_*.doc (supplied on CD).

4.2 Position Commands

Position command data coming into the Nano-Drive® from an attached computer is read by the DAC and applied to the nanopositioner at the chosen (or default) rate. Since the ADC and DAC have independent clocks, the read/write rates do not have to be the same. If the DAC rate is changed from the default value, the new DAC rate will remain in effect until it is changed again or until a data logging procedure is initiated (see below).

It is important to note that a latency exists between the time when a computer program issues a position command and the time when that command is communicated to the Nano-Drive® via the USB interface. The Windows operating system "talks to" each USB port on a regular 1 millisecond interval. This access timing can become important if the nanopositioner is controlled in a point-by-point manner and the desired motion is fairly fast (roughly 100Hz or faster). Fast motions should be controlled and monitored using the built-in data logging facility.

4.3 Reading Position

Under normal operation, the position of each axis is constantly recorded at the chosen (or default) ADC rate. Internal memory in the Nano-Drive® is used to hold 30 position values which are accessed and averaged together in the MADLib.dll software. As each new ADC value is stored, the oldest value is removed from memory to maintain a buffer of only 30 values. This process reduces the effects of random noise and produces highly reliable position data. It should

be noted, however, that reducing the chosen ADC rate too much can create an apparent “phase shift” in the position data since the total time span of all 30 values may be excessive. Leaving the ADC at the default rate (highest speed) is probably the best choice for most applications. If the ADC rate is changed from the default value, the new ADC rate will remain in effect until it is changed again or until a data logging procedure is initiated (see below).

4.4 Built-In Data Logging

The Nano-Drive® controller has internal memory to store up to 10,000 position command values and up to 10,000 position sensor values. Rather than reading/writing a single position value at a time (as described above), the USB interface can be used to read/write 10,000 data values to/from these internal memory locations. This method of data logging eliminates the 1ms/point Windows latency which limits nanopositioner data rates when the USB interface reads/writes a single point at a time. Software trigger commands initiate the flow of data in and out of the ADC/DAC circuits. The ADC and DAC clock rates are independently selectable from the range of values from 5ms/point to 33(s/point. After completing a data logging procedure, the Nano-Drive® reverts back to the default ADC/DAC rates and resumes the normal monitoring of the position by reacquiring a new array of 30 position values (as described above). A new single position read is available 3ms after completing a data logging procedure – this time is required to fill the 30 data point array.

4.5 ISS (Integrated Scan Synchronization) Option

The ISS option is a generalized Integrated Scan Synchronization for use with cameras, single photon counting boards and similar devices. This option provides TTL-compatible pulses used to digitally synchronize a Nano-Drive event, such as reading the X-axis position, with another external device, such as a single photon counting board.

The triggering event (read, write, etc.) of the pixel clock and three additional TTL outputs (labeled line clock, frame clock and aux clock) can be configured using the USB interface. Alternatively, clocks can be generated under the software control of the user, providing a way to synchronize the line and frame timing of cameras and single photon counting boards. Section 4.7.5 describes a provided LabView example, SawScan(Two axis + ISS option).vi, which uses the ISS option.

4.6 Nano-Route®3D software

Nano-Route®3D is a LabVIEW based motion control and data acquisition program that allows users to command our USB-enabled Nano-Drive® controllers. The software is shipped only with USB-enabled controllers and is provided on the same USB flash drive as the manuals. The software is provided as an executable and as LabVIEW VI's. Nano-Route®3D VI's incorporate many of the LabView examples described in section 4.7 and allow users to incorporate functionality of the Nano-Route®3D software into their own LabVIEW based software. The LabView run time engine is included in the "LabView Executables" folder and must be installed to run the Nano-Route®3D executable. Users must have LabView 2012 or later to run the VI's. An extensive tutorial document for Nano-Route®3D is provided with the software. Videos showing the use of Nano-Route®3D are available on YouTube[1][2].

Upon start-up Nano-Route®3D identifies the attributes of the attached Nano-Drive® controller, the user then has the option to generate and run routes based on preloaded parameters or custom entered parameters. This capability allows both novice and experienced users to quickly generate customized motion control, including complex multi-axis routes. Routes can be generated via simple "go to" commands or more sophisticated point or time based functions. Motion commands to the Nano-Drive® controller can be issued via single write mode or waveform acquisition mode (single download of all data points) depending on application requirements. A graphical display previews the generated command route and may be edited prior to the route being run. During execution of a route, Nano-Route®3D retains the ADC (sensor), and DAC (command) data retrieved from the Nano-Drive® controller. This information is displayed graphically in a separate window. The graphical representation can be customized and the information exported for analysis.

Features

- Easy to use graphical interface for both novice and experienced users
- Live position tracking
- Point and time based function paths
- Complex multi-axis route creation

[1] Mad City Labs Nano-Route®3D software for nanopositioning systems. https://youtu.be/MsdqcZJfe-g

[2] Generating a Z-Stack using Mad City Labs Nano-Route®3D software. https://youtu.be/KWfAGtFcDk0

- Control over pixel, line and frame clocks to facilitate complex path creation.
- Route preview window
- Graphical display of executed route (commanded and actual).
- Position information fully exportable
- LabVIEW compatible. Can be used “as is” or expanded by additional user generated programming.

4.7 LabView Examples

Several examples of software control using LabView have been installed in the LabView Examples folder of the root directory, which by default is C:\Program Files\Mad City Labs\NanoDrive. Documentation for each VI can be found on its Block Diagram and/or Front Panel. The supplied VIs are outlined below along with a brief description.

4.7.1 Common VIs

- ErrorLookup.vi translates error codes into a readable form. It can be used as a SubVI.
- GetCalibration.vi returns the calibration in microns (µm) of a given axis. It can be used as a SubVI.
- Nano-Drive Attached.vi returns a True or False value based on whether the Nano-Drive® is attached.
- RampGenerator.vi generates a linear ramp between two positions with a specified number of points. It is used as a SubVI in all of the scanning examples.
- ReadWriteN.vi reads the current position, writes a new position, then waits. Works well as a SubVI within scanning programs.
- StandardInit.vi initializes a Nano-Drive® so it can be used by software. Used as a SubVI
- StandardRelease.vi frees an initialized Nano-Drive® so it can be used by other programs. Used as a SubVI.
- WriteReadN.vi writes a position, waits, then reads the current position. Works well as a SubVI within scanning programs.

4.7.2 Device Information Examples

- Multiple Nano-Drives® Information.vi provides information on all available attached Nano-Drives®.
- Nano-Drive Information.vi displays information about a single Nano-Drive®.

4.7.3 Handle Management Examples

- GetHandleBySerial.vi initializes the Nano-Drive® that has the specified serial number.
- GetHandleByType.vi initializes the Nano-Drive® that has the specified type.

4.7.4 Multiple Device Examples

- Distinguished by serial.vi controls two separate Nano-Drives® simultaneously using the serial number to distinguish between the two.
- Distinguished by type.vi controls two separate Nano-Drives® simultaneously using the type of the Nano-Drive® to distinguish between the two.
- SingleControllerTwoStages.vi is used to control two MTA2 stages from a single controller.

4.7.5 Scan Examples

- SawScan(One axis).vi steps the X axis through a specified linear ramp and upon completion returns the X axis to its starting position.
- SawScan(Two axis).vi performs a scan such that for each step in a specified linear ramp on the Y axis the X axis is completely stepped through a specified linear ramp and upon completion returned to its starting position.
- SawScan(Two axis + ISS option).vi implements the same scan as SawScan(Two axis).vi and in addition generates the clock pulses needed for SPC applications. With the ISS option, the pixel clock is generated for every read of the X-axis. This example shows how to generate Line clocks and Frame clocks via software.
- TriangleScan(Two axis).vi performs a scan such that for each step in a specified linear ramp on the Y axis the X axis is completely stepped through a specified linear ramp and upon completion the X axis remains in its position and has its specified linear ramp inverted.

4.7.6 Simple Device Movement Examples

- Nano_Monitor.vi uses a LabView function generator to generate commonly used waveforms (such as sine, square, triangle, and sawtooth waveforms) on a single axis.
- NanoXYZ_Monitor.vi uses a LabView function generator to generate commonly used waveforms (such as sine, square, triangle, and sawtooth waveforms) on all axes.
- Read.vi reads the position of the nanopositioner in a continuous loop

4.7.7 TipTilt Z Examples

The example VI's described in this section are only relevant to nanopositioning systems with tip and tilt enabled.

- Align.vi provides three command inputs labeled as θ_A, θ_B, θ_C, which can be used to control each actuator independently.
- TipTilt Z.vi provides three command inputs; Center (µm), θ_X(mrad), and θ_Y(mrad) to control the orientation of the Z plane. Each of these commands produces an independent motion. The inputs are coerced by the function calls such that each command only affects one aspect of the planes orientation, i.e. a change in θ_X will not change θ_Y. This VI displays the current values for:
  - Center, θ_X, θ_Y, θ_A, θ_B, and θ_C
  - Height and width of the actuator triangle
  - Maximum and minimum achievable θ_X and θ_Y.
- XY with TipTilt Z.vi adds control of the X and Y axes to the attributes of TIP TILT Z.VI described above (5 axis).
- XYZ with TipTilt Z.vi adds control of the Theta Z axis to the attributes of XY with TIP TILT Z.VI described above (6 axis).

4.7.8 Waveform Acquisition Examples

- LoadWaveform (subVI).vi stores position commands in the Nano-Drive® memory. This VI triggers the Nano-Drive™ to use these command positions immediately
- LoadWaveformSetupTrigger.vi stores position commands in the Nano-Drive® memory with one function call and triggers the Nano-Drive® to use them with a another function call.
- ReadWaveform(subVI).vi can read up to 1000 consecutive position sensor values from the Nano-Drive® internal memory. This VI informs the Nano-Drive® to immediately begin recording data and the number (points) and rate at which data should be logged.
- ReadWaveformSetupTrigger.vi describes a two step process of describing how to log data (as described immediately above) and then triggering the Nano-Drive® to begin recording data.
- WaveformAcquisitionSynchronous.vi describes how to trigger the Nano-Drive® to use stored position control values and record consecutive position sensor values synchronously.

- WaveformAcquisitionSynchronous (subVI).vi is the same as the vi described above but in a subVI format for easier integration into user written LabView programs.
- Multi Axis Waveform Acquisition.vi provides an example of how to setup and trigger a multi-axis waveform.

4.7.9 Other

- NanoGauge.vi reads the encoder within a continuous loop.
- Raster Scan.vi provides step-by-step instructions for setting up and performing a multi-axis raster scan.
- Step Response (16 bit).vi reads the nanopositioner at a fast rate while the nanopositioner moves to a position. Cannot be used with 20 bit USB controllers.
- Step Response (20 bit).vi reads the nanopositioner at a fast rate while the nanopositioner moves to a position. Cannot be used with 16 bit USB controllers.
- Keyboard NanoDrive Control.vi maps keyboard keys to nanopositioner motions to allow keyboard control of the nanopositioner.
- GamePad NanoDrive Control maps gamepad buttons to nanopositioner motions to allow gamepad control of the nanopositioner.
- Cfocus.vi locks and unlocks focus. Allows the stage to be displaced while maintaining focus lock.

4.8 Programming in other languages

Use the Madlib.h, Madlib.lib, and Madlib.dll to develop software for the Nano-Drive® USB interface. All three files can be found in the root directory, which by default is C:\Program Files\Mad City Labs\NanoDrive.

5 OPERATING THE NANO-DRIVE® VIA THE 20 BIT USB INTERFACE (OPTION)

This section describes the operation of the Nano-Drive controller via the USB 20 bit interface option operating under Microsoft Windows.

The 20 bit USB interface on the Nano-Drive® controls up to three axes simultaneously, depending on the product purchased. The position of each axis is controlled independently. Each axis has a dedicated 20 bit DAC and 24 bit ADC that are directly accessed by the USB interface. Inside the Nano-Drive®, an 8051 microprocessor controls the ADC and DAC. Rate limitations for single read/write actions on the ADC/DAC are listed in Table 4 below. For information about implementation of read/write actions see sections 5.2-5.4. Communication with the host PC is controlled by the installed USB driver, and the driver is accessed by function calls to the Madlib.dll library. The position, in micrometers, of the nanopositioner can be changed and monitored under software control by using the supplied dynamically linked library (DLL).

Effective ADC Rate	***DAC Rate***
20ms/point, 17ms/point, 10ms/point, 2ms/point, 1ms/point, 500µs/point**,** 267µs/point **(2ms/point is the default rate)**	5ms/point to 167µs/point (in 83.33ns steps) **(2ms/point is the default rate)**

Table 4: ADC/DAC rate limitations for single read/write actions

5.1 Introduction to MADLib.dll

Windows allows programs to link to function calls in either static or dynamic libraries. For a program to link to a static library, it must be incorporated when the program is compiled, while for a dynamically linked library, the program is linked to the library when it is run. Programs written in LabView or Visual Studio (C++, C#, VB etc.) can make function calls to dynamically linked libraries such as MADLib.dll. In LabView, these function calls to an external DLL are referred to as “call library functions”. The user may wish to study the many LabView examples provided and read the LabView documentation to get a better understanding of how to used “call library functions”. In order to view the LabView examples you must have LabView installed (not supplied).

The USB driver provided with the Nano-Drive® is designed to mediate communications between Windows and the Nano-Drive® . Communication between the USB driver and the user is

controlled by MADLib.dll When a user wishes to access a USB device such as the Nano-Drive®, a “handle” to the device must be opened in Windows as a means for the user and Windows to identify a specific device the user wishes to communicate with. In this sense, a “handle” is simply a unique identifier.

All programs should be of the form:

<...>

Initialize handle(s).

Program portion that relies on the Nano-Drive®(s).

Release handle(s).

<...>

5.1.1 Using a single Nano-Drive® controller

Using a single controller three functions are required to properly manage handles:

- MCL_InitHandle will obtain control of the controller. Should a LabView program stop/crash/finish prior to releasing the handle subsequent calls to MCL_InitHandle will fail since the DLL still has the handle.
- MCL_InitHandleOrGetExisting will obtain control of the controller. This function will return control of the controller even if the LabView program did not properly release the handle.
- MCL_ReleaseHandle will relinquish control of the controller.

5.1.2 Using multiple Nano-Drive® controllers

Using multiple Nano-Drives MCL_InitHandle and MCL_GrabHandle may have ambiguous results. MCL_InitHandle will obtain a handle to one of the Nano-Drive® (s) attached. MCL_GrabHandle may be used to differentiate between different classes of Nano-Drive® (s), however, if multiple devices of the same class are attached the result will be ambiguous. To deal with this ambiguity each Nano-Drive® has one unique feature, namely, its serial number. MCL_GrabAllHandles may be used to obtain handles to all Nano-Drive® (s) attached but not yet under control. MCL_GetHandleBySerial may be used to search all of the attached and controlled Nano-Drive® (s) for a Nano-Drive® with a matching serial number.

More detailed documentation on these functions can be found in the document Madlib_*_*.doc (supplied on CD).

*****Read the documentation for more information on the handle management functions.*****

5.2 Position Commands

Position command data coming into the Nano-Drive® from an attached computer is read by the DAC and applied to the nanopositioner at the chosen (or default) rate. Since the ADC and DAC have independent clocks, the read/write rates do not have to be the same. If the DAC rate is changed from the default value, the new DAC rate will remain in effect until it is changed again or until a data logging procedure is initiated (see below).

It is important to note that a latency exists between the time when a computer program issues a position command and the time when that command is communicated to the Nano-Drive® via the USB interface. The Windows operating system "talks to" each USB port on a regular 1 millisecond interval. This access timing can become important if the nanopositioner is controlled in a point-by-point manner and the desired motion is fairly fast (roughly 100Hz or faster). Fast motions should be controlled and monitored using the built-in data logging facility.

5.3 Reading Position

Under normal operation, the position of each axis is constantly recorded at the chosen (or default) effective ADC rate. Leaving the ADC at the default effective rate is probably the best choice for most applications. If the ADC rate is changed from the default value, the new ADC rate will remain in effect until it is changed again or until a data logging procedure is initiated (see below).

5.4 Built-In Data Logging

The Nano-Drive® controller has internal memory to store up to 6666 position command values and up to 6666 position sensor values. Rather than reading/writing a single position value at a time (as described above), the USB interface can be used to read/write 6666 data values to/from these internal memory locations. This method of data logging eliminates the 1ms/point Windows latency which limits nanopositioner data rates when the USB interface reads/writes a single point at a time. Software trigger commands initiate the flow of data in and out of the ADC/DAC circuits. The ADC and DAC clock rates are independently selectable from the range of values shown in Table 4. After completing a data logging procedure, the Nano-Drive® reverts back to the default ADC/DAC rates and resumes the normal monitoring of the position.

5.5 ISS option

The ISS option is a generalized Integrated Scan Synchronization for use with cameras, single photon counting boards and similar devices. This option provides TTL-compatible pulses used to digitally synchronize a Nano-Drive event, such as reading the X-axis position, with another external device, such as a single photon counting board.

The triggering event (read, write, etc.) of the pixel clock and three additional TTL outputs (labeled line clock, frame clock and aux clock) can be configured using the USB interface. Alternatively, clocks can be generated under the software control of the user, providing a way to synchronize the line and frame timing of cameras and single photon counting boards. Section 5.7.5 describes a provided LabView example, SawScan(Two axis + ISS option).vi, which uses the ISS option.

5.6 Nano-Route®3D software

Nano-Route®3D is a LabVIEW based motion control and data acquisition program that allows users to command our USB-enabled Nano-Drive® controllers. The software is shipped only with USB-enabled controllers and is provided on the same USB flash drive as the manuals. The software is provided as an executable and as LabVIEW VI's. Nano-Route®3D VI's incorporate many of the LabView examples described in section 4.7 and allow users to incorporate functionality of the Nano-Route®3D software into their own LabVIEW based software. The LabView run time engine is included in the "LabView Executables" folder and must be installed to run the Nano-Route®3D executable. Users must have LabView 2012 or later to run the VI's. An extensive tutorial document for Nano-Route®3D is provided with the software. Videos showing the use of Nano-Route®3D are available on YouTube[3][4].

Upon start-up Nano-Route®3D identifies the attributes of the attached Nano-Drive® controller, the user then has the option to generate and run routes based on preloaded parameters or custom entered parameters. This capability allows both novice and experienced users to quickly generate customized motion control, including complex multi-axis routes. Routes can be generated via simple "go to" commands or more sophisticated point or time based functions. Motion commands to the Nano-Drive® controller can be issued via single write mode or

[3] Mad City Labs Nano-Route®3D software for nanopositioning systems. https://youtu.be/MsdqcZJfe-g

[4] Generating a Z-Stack using Mad City Labs Nano-Route®3D software. https://youtu.be/KWfAGtFcDk0

waveform acquisition mode (single download of all data points) depending on application requirements. A graphical display previews the generated command route and may be edited prior to the route being run. During execution of a route, Nano-Route®3D retains the ADC (sensor), and DAC (command) data retrieved from the Nano-Drive® controller. This information is displayed graphically in a separate window. The graphical representation can be customized and the information exported for analysis.

Features

- Easy to use graphical interface for both novice and experienced users
- Live position tracking
- Point and time based function paths
- Complex multi-axis route creation
- Control over pixel, line and frame clocks to facilitate complex path creation.
- Route preview window
- Graphical display of executed route (commanded and actual).
- Position information fully exportable
- LabVIEW compatible. Can be used “as is” or expanded by additional user generated programming.

5.7 LabView Examples

Several examples of software control using LabView have been installed in the LabView Examples folder of the root directory, which by default is C:\Program Files\Mad City Labs\NanoDrive. Documentation for each VI can be found on its Block Diagram and/or Front Panel. The supplied VIs are outlined below along with a brief description.

5.7.1 Common VIs

- ErrorLookup.vi translates error codes into a readable form. It can be used as a SubVI.
- GetCalibration.vi returns the calibration in microns (µm) of a given axis. It can be used as a SubVI.
- Nano-Drive Attached.vi returns a True or False value based on whether the Nano-Drive® is attached.
- RampGenerator.vi generates a linear ramp between two positions with a specified number of points. It is used as a SubVI in all of the scanning examples.

- ReadWriteN.vi reads the current position, writes a new position, then waits. Works well as a SubVI within scanning programs.
- StandardInit.vi initializes a Nano-Drive® so it can be used by software. Used as a SubVI
- StandardRelease.vi frees an initialized Nano-Drive® so it can be used by other programs. Used as a SubVI.
- WriteReadN.vi writes a position, waits, then reads the current position. Works well as a SubVI within scanning programs.

5.7.2 Device Information Examples

- Multiple Nano-Drives® Information.vi provides information on all available attached Nano-Drives®.
- Nano-Drive Information.vi displays information about a single Nano-Drive®

5.7.3 Handle Management Examples

- GetHandleBySerial.vi initializes the Nano-Drive® that has the specified serial number.
- GetHandleByType.vi initializes the Nano-Drive® that has the specified type.

5.7.4 Multiple Device Examples

- Distinguished by serial.vi controls two separate Nano-Drives® simultaneously using the serial number to distinguish between the two.
- Distinguished by type.vi controls two separate Nano-Drives® simultaneously using the type of the Nano-Drive® to distinguish between the two.
- SingleControllerTwoStages.vi is used to control two MTA2 stages from a single controller.

5.7.5 Scan Examples

- SawScan(One axis).vi steps the X axis through a specified linear ramp and upon completion returns the X axis to its starting position.
- SawScan(Two axis).vi performs a scan such that for each step in a specified linear ramp on the Y axis the X axis is completely stepped through a specified linear ramp and upon completion returned to its starting position.
- SawScan(Two axis + ISS option).vi implements the same scan as SawScan(Two axis).vi and in addition generates the clock pulses needed for SPC applications. With the ISS

option, the pixel clock is generated for every read of the X-axis. This example shows how to generate Line clocks and Frame clocks via software.

- TriangleScan(Two axis).vi performs a scan such that for each step in a specified linear ramp on the Y axis the X axis is completely stepped through a specified linear ramp and upon completion the X axis remains in its position and has its specified linear ramp inverted.

5.7.6 Simple Device Movement Examples

- Nano_Monitor.vi uses a LabView function generator to generate commonly used waveforms (such as sine, square, triangle, and sawtooth waveforms) on a single axis.
- NanoXYZ_Monitor.vi uses a LabView function generator to generate commonly used waveforms (such as sine, square, triangle, and sawtooth waveforms) on all axes.
- Read.vi reads the position of the nanopositioner in a continuous loop.

5.7.7 TipTilt Z Example

The example VI's described in this section are only relevant to nanopositioning systems with tip and tilt enabled.

- Align.vi provides three command inputs labeled as θ_A, θ_B, θ_C, which can be used to control each actuator independently.
- TipTilt Z.vi provides three command inputs; Center (µm), θ_X(mrad), and θ_Y(mrad) to control the orientation of the Z plane. Each of these commands produces an independent motion. The inputs are coerced by the function calls such that each command only affects one aspect of the planes orientation, i.e. a change in θ_X will not change θ_Y. This VI displays the current values for:
  - Center, θ_X, θ_Y, θ_A, θ_B, and θ_C
  - Height and width of the actuator triangle
  - Maximum and minimum achievable θ_X and θ_Y.
- XY with TipTilt Z.vi adds control of the X and Y axes to the attributes of TIP TILT Z.VI described above (5 axis).
- XYZ with TipTilt Z.vi adds control of the Theta Z axis to the attributes of XY with TIP TILT Z.VI described above (6 axis).

5.7.8 Waveform Acquisition Examples

- LoadWaveform (subVI).vi stores position commands in the Nano-Drive® memory. This VI triggers the Nano-Drive™ to use these command positions immediately
- LoadWaveformSetupTrigger.vi stores position commands in the Nano-Drive® memory with one function call and triggers the Nano-Drive® to use them with a another function call.
- ReadWaveform(subVI).vi can read up to 1000 consecutive position sensor values from the Nano-Drive® internal memory. This VI informs the Nano-Drive® to immediately begin recording data and the number (points) and rate at which data should be logged.
- ReadWaveformSetupTrigger.vi describes a two step process of describing how to log data (as described immediately above) and then triggering the Nano-Drive® to begin recording data.
- WaveformAcquisitionSynchronous.vi describes how to trigger the Nano-Drive® to use stored position control values and record consecutive position sensor values synchronously.
- WaveformAcquisitionSynchronous (subVI).vi is the same as the vi described above but in a subVI format for easier integration into user written LabView programs.
- Multi Axis Waveform Acquisition.vi provides and example of how to setup and trigger a multi-axis waveform.

5.7.9 Other

- NanoGauge.vi reads the encoder within a continuous loop.
- Raster Scan.vi provides step-by-step instructions for setting up and performing a multi-axis raster scan.
- Step Response (16 bit).vi reads the nanopositioner at a fast rate while the nanopositioner moves to a position. Cannot be used with 20 bit USB controllers.
- Step Response (20 bit).vi reads the nanopositioner at a fast rate while the nanopositioner moves to a position. Cannot be used with 16 bit USB controllers.
- Keyboard NanoDrive Control.vi maps keyboard keys to nanopositioner motions to allow keyboard control of the nanopositioner.
- GamePad NanoDrive Control maps gamepad buttons to nanopositioner motions to allow gamepad control of the nanopositioner.
- Cfocus.vi locks and unlocks focus. Allows the stage to be displaced while maintaining focus lock.

5.8 Programming in other languages

Use the Madlib.h, Madlib.lib, and Madlib.dll to develop software for the Nano-Drive® USB interface. All three files can be found in the root directory, which by default is C:\Program Files\Mad City Labs\NanoDrive.

6 TROUBLE SHOOTING

The Nano-Drive® is a high precision instrument and therefore must be operated in a quiet, both physical and electrical, environment. Both electrical and vibration noise can be picked up in the Nano-Drive® and the nanopositioning stage. This noise can be observed on the HV/10 BNC output. The nanopositioning system is a high voltage device and is designed for operation in a dry (non-condensing) environment.

The tables on the following pages describe a range of possible problems and their solutions. Please contact Mad City Labs or your local authorized dealer (visit www.madcitylabs.com for information) for technical support.

Problem	Cause	Solution
The stage does not move. OR The range of motion of the stage is incorrect OR The motion is incorrect	Sensor offset adjustment is not set correctly	See section 2.3 for instructions on adjusting the sensor offset.
	External stage obstruction	Check the area around the nanopositioning stage with particular attention to sample holders, fasteners or external hardware that may be restricting the motion of the stage.
	Internal stage obstruction. The most likely cause of internal stage obstructions are dried salts resulting from saline spills.	Contact Mad City Labs for an RMA number. Internal stage obstructions can only be removed by authorized service personnel.
	High voltage is not being supplied to piezoactuators AND/OR The sensors are faulty	<u>For analog input Nano-Drive® controllers</u> Follow the instructions in the document "Nanopositioning diagnostic – analog". Send the resulting table of values to Mad City Labs. <u>USB enabled Nano-Drive® controllers</u> Run the diagnostic software and send resulting file to Mad City Labs. Instructions for running the diagnostic software are in the document "Nanopositioning diagnostic software".
	Wrong analog input voltage is applied	Standard systems use a 0-10V analog input signal. The analog input voltage range is shown on the certificate of performance delivered with your nanopositioning system. If you do not have the certificate, please contact Mad City Labs with your serial number. Serial numbers can be found either on the stage or the rear panel of the controller.
	Inputs from both the USB and analog voltage connectors	The digital (USB) and analog input voltages are summed. Therefore it is essential for the user to check that both inputs are at zero displacement before beginning any commands.

Problem	Cause	Solution
Low frequency (1-5Hz) noise	Acoustic noise, e.g. foot traffic, building noise.	Vibration isolation of the nanopositioning stage.
60, 120Hz noise	Ground loop interference caused by a slight difference in the ground potentials of the nanopositioning stage and the Nano-Drive® controller.	Electrically isolate the nanopositioning stage. More information on ground loops can be found in the accompanying nanopositioning stage manual.
The stage makes a crackling noise and then does not move.	Short circuit in the stage	Request an RMA number. The stage and controller must be returned to Mad City Labs for repair
The stage is unstable or oscillates	Ground loop	See “ground loop” section in the nanopositioning stage manual.
	Large mass placed on stage	Reduce mass on the stage or contact Mad City Labs support for assistance.
The USB port LED light is not lit	USB port is not active	Cycle the power to the Nano-Drive® controller. If the light still does not come on and the device is not recognized by your computer, please contact MCL for an RMA number
I cannot find the manuals		The manuals and any available software are automatically installed on your PC when you insert the Manual CD delivered with the nanopositioning system.
I cannot find the Nano-Route®3D software		The Nano-Route®3D software is automatically installed when you first insert the Manual CD. Nano-Route®3D software is only shipped with USB enabled Nano-Drive® controllers.
The Nano-Route®3D software does not run	LabVIEW runtime engine is not installed or the user is running an older version of LabVIEW	The LabVIEW runtime engine must be installed manually from the CD. It can also be downloaded from the National Instruments website. You must be running LabVIEW 2012 or later to use the VIs.

Problem	Cause	Solution
The software stopped working.	Change in operating system or PC.	Older versions of the software are not compatible with 32 bit and 64 bit Windows 7 and newer. New drivers exist and can be requested from Mad City Labs at no charge.