Panther Programmer

USER MANUAL

Version 1.05
February, 2024

Aledyne

Aledyne has made every attempt to ensure that the information in this document is accurate and complete. However, Aledyne assumes no responsibility for any errors, omissions, or for any consequences resulting from the use of the information included herein or the equipment it accompanies. Aledyne reserves the right to make changes in its products and specifications at any time without notice. Any software described in this document is furnished under a license or non-disclosure agreement. It is against the law to copy this software on magnetic tape, disk, or other medium for any purpose other than the licensee's personal use.

Aledyne Engineering, Inc.
P.O. Box 1530
Morgan Hill, CA. 95038

Tel: (408) 465-9825
Web: www.aledyne.com
E-Mail: support@aledyne.com

Acknowledgments:

USER MANUAL

TABLE OF CONTENTS

1.0 INTRODUCTION

1.0 INTRODUCTION

1.1 System Overview

The Panther 1-Wire® Programmer is a standalone programmer for Maxim Integrated 1-Wire® memory devices. The Panther Programmer has the capability of issuing a high voltage (12V) programming pulse that the Add-Only memory types require for programming, which no other programmer on the market provides. It allows for reading and writing the contents of numerous types of 1-Wire® memory devices. It also has the capability of reading the unique ROM code (unique ID) of all 1-Wire memory devices. A generic mode can also be configured which allows customization of communication parameters of devices not on the supported list.

The Panther Programmer can operate in stand-alone mode using a MMC/SD card or with a host computer through a virtual serial port. A LabVIEW driver is also available on the LabVIEW Tools Network which implements the entire commands set for the Programmer and allows for easy integration to LabVIEW and TestStand applications for design, development, and production.

The Panther Programmer is a perfect solution for development and production of disposable devices in medical equipment, smart cables, automotive, aerospace, consumer, and any other application using a 1-Wire® memory device. The Panther Programmer will help to reduce Time-To-Market and increase the quality of the end product.

1.2 Supported Devices

The following 1-Wire® memory devices are supported. It is possible that future firmware releases will include additional devices. It is also possible to configure generic memory device properties that may allow the Panther Programmer to work with unlisted devices. Contact support if your device is not listed for additional details on configuring for your device or future firmware upgrades.

- DS2401
- DS2406
- DS2430
- DS2431
- DS2433
- DS2502
- DS2502-E64
- DS2505
- DS2506
- DS28E04-100
- DS28E05*
- DS28E07
- DS28EC20
- DS28E80

- DS2434
- DS2436
- Memory iButtons (DS1971, DS1973, DS1992, DS1993, DS1995, DS1996)

*device only operates in Overdrive mode, which requires very tight time constraints. Although it may work, connecting to devices using the auxiliary connecter with an extended piece of wiring may not be reliable and is not supported by Aledyne.

1.3 Stand-alone Programming

The Panther Programmer is a great tool to use in early development in stand-alone mode to write or read a set of data on/from a connected memory device. To do this, a configuration file is loaded onto the Panther Programmer SD Card which contains configurations for the type of memory device, the address to read from or write to, the type of access (read/write/verify), and the file to read or write to/from. This is a perfect solution if you are experimenting with changing data to simulate different conditions. The user interface on the Panther Programmer can be used to quickly switch between configurations on the SD Card. More than 20 configurations can be setup at one time so that the user can quickly switch between them on the fly to either communicate with different devices or program different datasets onto the connected device.

1.4 Stand-alone Verification

The Panther Programmer can also be used as a verification only step to ensure the proper data is stored on the memory device following a programming step at a different production station or facility. The data file can be placed on the SD Card and, through a configuration, the Panther programmer will read the contents of the memory chip at the desired location (and memory size) and will compare to the data in the specified file. If the contents do not match, an error will be produced.

1.5 Production Programming (Stand-alone)

The Panther Programmer is a great fit for production programming of a 1-Wire® memory device with the same data. This allows the operator to use the programmer without needing a PC or special software. Simply load the data file on the SD card, setup the configuration file to read from the data file and write to the desired memory location on the device. A verify step can also be added so that the data is read back by the Programmer after writing to confirm it was successful.

1.6 Production Programming (Remote)

In remote mode, the Panther Programmer can be integrated to any automated production process through the serial interface described in Section X. There is also a free LabVIEW toolkit available to allow for easy integration to automated LabVIEW and TestStand applications. This mode allows the programmer to write unique data to each specific 1-Wire® memory device during production. This is useful if a

specific device serial number needs to be written on each unit. Alternatively, the unique ROM code can also be read back and recorded to a product database during production.

We also offer an OEM version of the programmer with PCB only (no enclosure or display) if the customer wishes to integrate the PCB into their own manufacturing test fixture and communicate with it entirely over the USB interface.

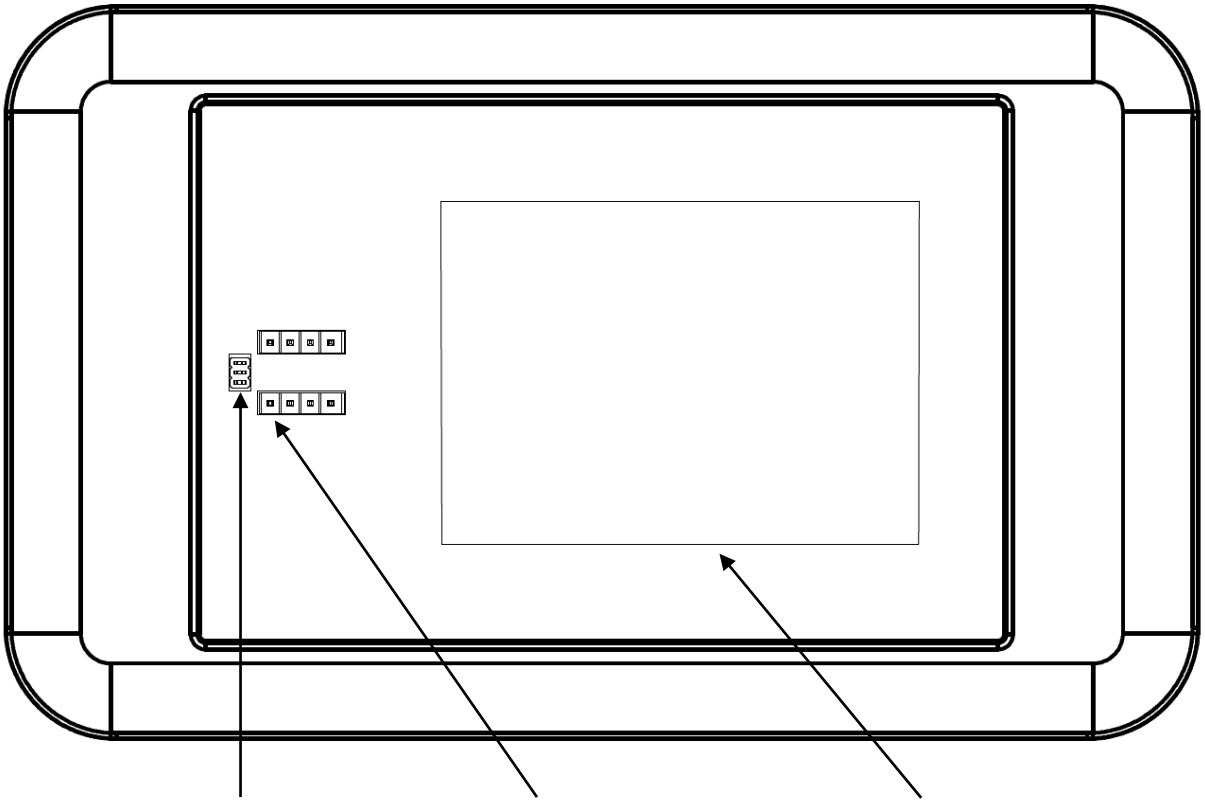
2.0 SYSTEM SUMMARY

2.0 SYSTEM SUMMARY

This chapter will describe the Panther Programmer's user interface, operating modes, and basic steps for configuration.

To start using the Panther Programmer you need to make a memory configuration file. This may already be shipped with your device.

2.1 Front Panel Interface



1. TO-92 1-Wire Connector

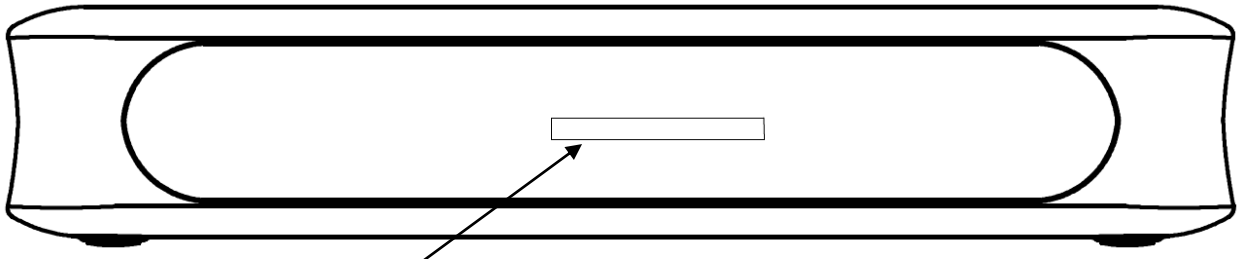
Pin 1: PIO (Ground or 3.3V)
Pin 2: Data
Pin 3: Ground

2. SO Adapter 1-Wire Connector

Pin 1: NC	Pin 5: Ground
Pin 2: Ground	Pin 6: Data
Pin 3: Data	Pin 7: Ground
Pin 4: Ground	Pin 8: NC

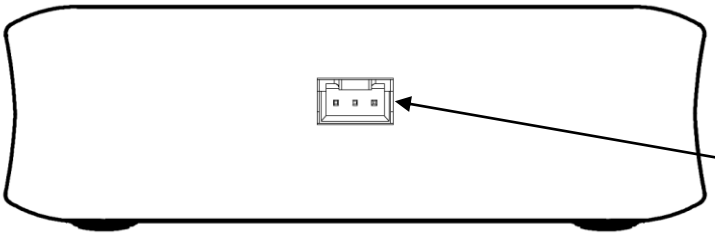
3. Touchscreen LCD

2.2 Side Panel Interface



4. SD Card Slot

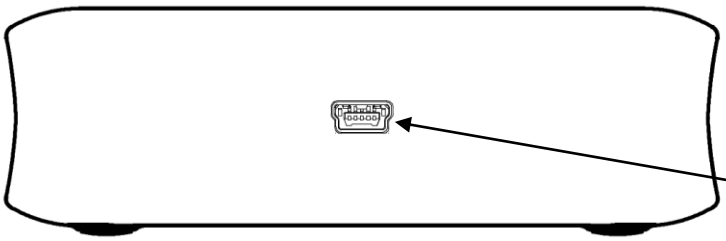
2.3 Top Panel Interface



5. Auxiliary 1-Wire Connector

Pin 1: PIO (Ground or 3.3V)
Pin 2: Data
Pin 3: Ground

2.4 Bottom Panel Interface



6. Mini USB Connector

2.5 Description

1. TO-92 1-Wire Connector: Connector used for interfacing to TO-92 style 1-Wire memory devices. The 1-Wire memory device can be inserted in either direction.

2. SO Adapter 1-Wire Connector: Connector used for attaching a surface mount adapter to connect to SO-8 style 1-Wire memory devices. The adapter can be inserted in either direction.

3. Touchscreen LCD: User interface for standalone mode of operation.

4. SD Card Slot: Slot for the SD card used for storing data, configuration, and license files.

5. Auxiliary 1-Wire Connector: Provides a rigid connection to a test fixture for a permanent installation.

6. Mini USB Connector: The USB connector provides the unit with +5VDC power and also is used for remote control. If the unit is being operated in standalone mode, it can be plugged into a USB wall power adapter.

3.0 GETTING STARTED

3.0 GETTING STARTED

3.1 Setting up

Remove the SD card from the box and copy the license file (*license.txt*) received at time of purchase onto the SD card. This may already be included with your SD card. Edit the configuration file (*config.txt*) to fit your needs. Insert the SD card into the Panther Programmer and power up the Programmer by connecting to a computer with the included Mini USB cable, or connect to a USB Power adapter. The first time the unit is powered up, the license file will be read and copied to internal memory. It can be removed from the SD card after first power up if desired.

Attach a supported TO-92 memory device to the 3-Pin connector on the top as shown below. The orientation does not matter and the TO-92 device can be inserted either direction, unless if PIO is enabled, then the TO-92 device must be reverse from the orientation shown below.

TO-92 Memory Device Setup

If you are using a surface mount device, connect the appropriate adapter to the SO adapter headers on the top of the programmer. The orientation does not matter and the SO adapter can be inserted either direction.

SO-8 Memory Device Setup

4.0 USING PANTHER

4.0 USING PANTHER

4.1 Configuration File

The configuration file is stored on the root directory of the SD card and must be named "*config.txt*". This file defines the 1-Wire configurations that are allowed from the user interface. A confirmation defines attributes such as the 1-Wire device type, the action to perform (read/write/verify), the memory address to read from or write to, the size of memory to read, etc.

4.1.1 Attributes

name: The name of the configuration to display on the LCD (display only). This is limited to 30 characters.

command: The following commands are recognized:

READ – will read the data from the 1-Wire device and write it to the specified file.
WRITE – will write the data from the specified file to the 1-Wire device.
READ_VERIFY – will read the data from the 1-Wire device and compare it to the data in the specified file for correctness.
WRITE_READ_VERIFY – will first write the data from the specified file to the 1-Wire device. Then the memory location will be read back and compared with the specified file for correctness.
ROM – Will read the unique ROM code from the 1-Wire device and write it to the specified file.
READ_CYCLES – will read the remaining memory write cycles of the block corresponding to the specified address (DS28E80).
READ_PROTECTION – will read the protection status of the block corresponding to the specified address (DS28E80).
WRITE_PROTECTION – will write the protection status of the block corresponding to the specified address (DS28E80).

device: The type of 1-Wire memory device. The device ID is the same as the device family code as documented in the 1-Wire memory datasheet. The decimal value of the family code should be used. The following family codes are recognized:

0 – Generic (Used for undefined 1-Wire memory types)
1 – DS2401 (Silicon Serial Number)
18 – DS2406 (Dual Addressable Switch Plus 1Kb Memory)
20 – DS2430/DS2430A/DS1971 (256-Bit EEPROM)
45 – DS2431/DS28E07/DS1972 (1024-Bit EEPROM)
35 – DS2433/DS24B33/DS1973 (4Kb EEPROM)
9 – DS2502 (1Kb Add-Only Memory)
11 – DS2505 (16Kb Add-Only Memory)
15 – DS2506 (64Kb Add-Only Memory)
13 – DS28E05 (112 byte EEPROM)
137 – DS2502-E64 (1Kb Add-Only Memory)
74 – DS28E80 (248 byte Gamma Radiation Resistant 1-Wire Memory)
67 – DS28EC20 (20Kb Serial EEPROM)
28 – DS28E04-100 (4Kb EEPROM with PIO)
8 – DS1992 Memory iButton (1Kb EEPROM)

6 – DS1993 Memory iButton (4Kb EEPROM)
10 – DS1995 Memory iButton (16Kb EEPROM)
12 – DS1996 Memory iButton (64Kb EEPROM)
27 – DS2436 (Battery ID/Monitor Chip)
255 – DS2434 (Battery Identification Chip)
Note: Does not have Family device ID capabilities so it will be read as 0xFF when reading the ID

file: Defines the file name to read from (when performing a WRITE or WRITE_PROTECTION command) or the file to write to (when performing a READ, READ_PROTECTION, or READ_CYCLES command). This is an 8.3 filename (maximum of 8 characters for the name and 3 characters for the extension) and does not support directories. The file must be in the root directory of the SD card. For example: *dump.bin*

offset: Defines the decimal offset in bytes to read from or write to. If reading, the read operation will start at the specified offset. If writing, the write operation will start from the specified offset. For example: *64* defines reading or writing at byte 64 (0x40) of the connected 1-Wire memory device.

size: Defines the size of the data to read for read operations. This is only required for READ operations. For WRITE, READ_VERIFY, and WRITE_READ_VERIFY the size is automatically determined from the file size to write or verify against.

SNMode: Defines the read mode for the ROM read and READ commands. If this attribute is not present, the default mode is 0x00. The SNMode contains the following bitmasks:

Bit 0: If set, this enables append mode and each read of ROM or data will append the data to the same file on the SD card. If cleared, the specified file is deleted before each ROM or READ command.
Bit 1: If set, this enables ASCII mode and data is written to the SD card file in ASCII format. If cleared, data is written in binary format. Note: ASCII mode can only be enabled for ROM read.

For example, to append each ROM read to a single file in ASCII format add “SNMode=3” as follows:

```
#Read ROM ASCII Append
name=Read ROM
command=ROM
SNMode=3
file=ROM.bin
```

PIO: Defines if Pin 1 on the TO-92 connector and the Auxiliary connector is Ground or 3.3V. If this attribute is not present, the default mode is Ground. If set to 1, Pin 1 will be set to 3.3V and if set to 0, Pin 1 will be set to Ground. This can be used in order to place an external pull up resistor on the Data line to provide faster rise times if required or to apply power to a 1-wire device that has a separate power pin such as the DS2434 or DS2436. For example:
PIO=1 or PIO=0

The following attributes are used for a Generic memory device only:

type: The type of memory, either Add-Only or EEPROM, to set for the generic memory device. Add-Only type of devices require a high voltage (12V) programming pulse and typically have a scratchpad size of 1 byte. EEPROM type devices do not require high voltage programming and typically have a scratchpad size greater than 1 byte. Acceptable values are: “eeprom” or “addon”. For example:

type=eeprom

crcsize: The size of the CRC generator on the 1-Wire device in bits for the generic memory device. Acceptable values are: “8” or “16”. Anything else will disable the CRC check. For example:

crcsize=8

addrsize: The size of the address sent to the 1-Wire device in bits for the generic memory device. Acceptable values are: “8” or “16”. For example:

addrsize=16

spsize: Size in bytes to use for the scratchpad for the generic memory device. The scratchpad is a volatile portion of the memory where data is temporarily stored before writing to the non-volatile portion of memory. This should be set to the scratchpad size of the specific memory device or smaller. **Note: Only required for EEPROM Memory Type.**

spsize=8

spdelay: The delay in milliseconds to wait after issuing a copy scratchpad command for the generic memory device, which instructs the 1-Wire device to copy data from the scratchpad to the non-volatile memory location. Typically this must be at least 10ms but can vary from device to device. **Note: Only required for EEPROM Memory Type.**

spdelay=10

4.1.2 Commented Lines

To specify to the Panther Programmer to ignore a line in the configuration file, start the line with a ‘#’ character.

4.1.3 Generic Memory Device

If you would like to interface to a memory device that is not in the list of supported devices (see section 1.2), the generic memory device can be used to configure some of the communication protocol properties. In some cases this may work to interface with a non-supported device if the scratchpad size, address size, or crc size, for example need to be adjusted. In other cases, new 1-Wire memory devices may not work with the current firmware release of the Panther Programmer. Contact support@aledyne.com if you would like to request support for additional devices. This can provided for an additional fee.

4.1.4 Example

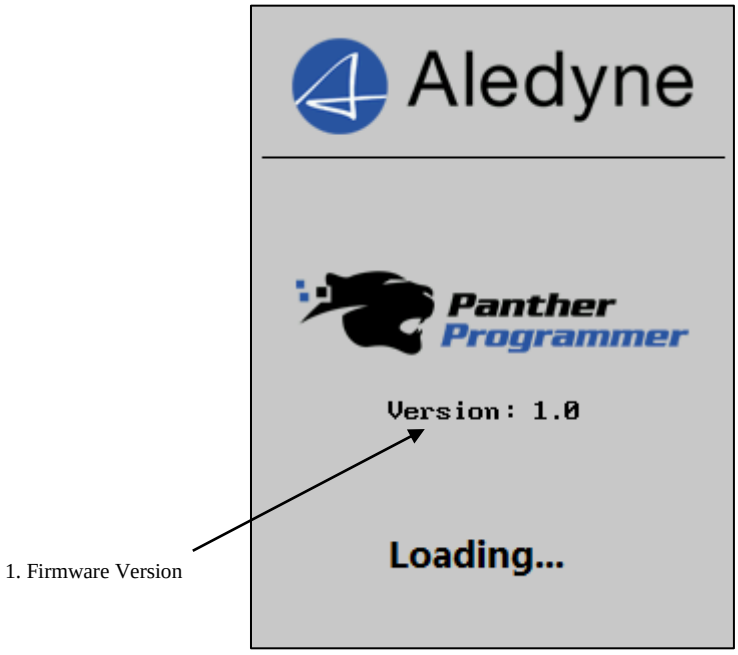
The following is an example of a configuration file with 7 configurations with 2 different devices (DS2433 and DS2505) selectable from the user interface.

```
#Configuration 1: Read ROM
name=Read One Wire ROM
command=ROM
file=ROM.bin

#Configuration 2: Write/Read/Verify 2433
name=Write/Read/Verify DS2433
command=WRITE_READ_VERIFY
device=35
```

```
file=data.bin
offset=0

#Configuration 3: Read/Verify 2433
name=Read/Verify DS2433
command=READ_VERIFY
device=35
file= data.bin
offset=0

#Configuration 4: Read 2433
name=Read DS2433
command=READ
device=35
file=dump.bin
offset=0
size=512

#Configuration 5: Write 2433
name=Write DS2433
command=WRITE
device=35
file= data.bin
offset=0

#Configuration 6: Read 2505
name=Read DS2505
command=READ
device=11
file=dump2505.bin
offset=0
size=2048

#Configuration 7: Write 2505
name=Write DS2505
command=WRITE
device=11
file= data2505.bin
offset=0
```

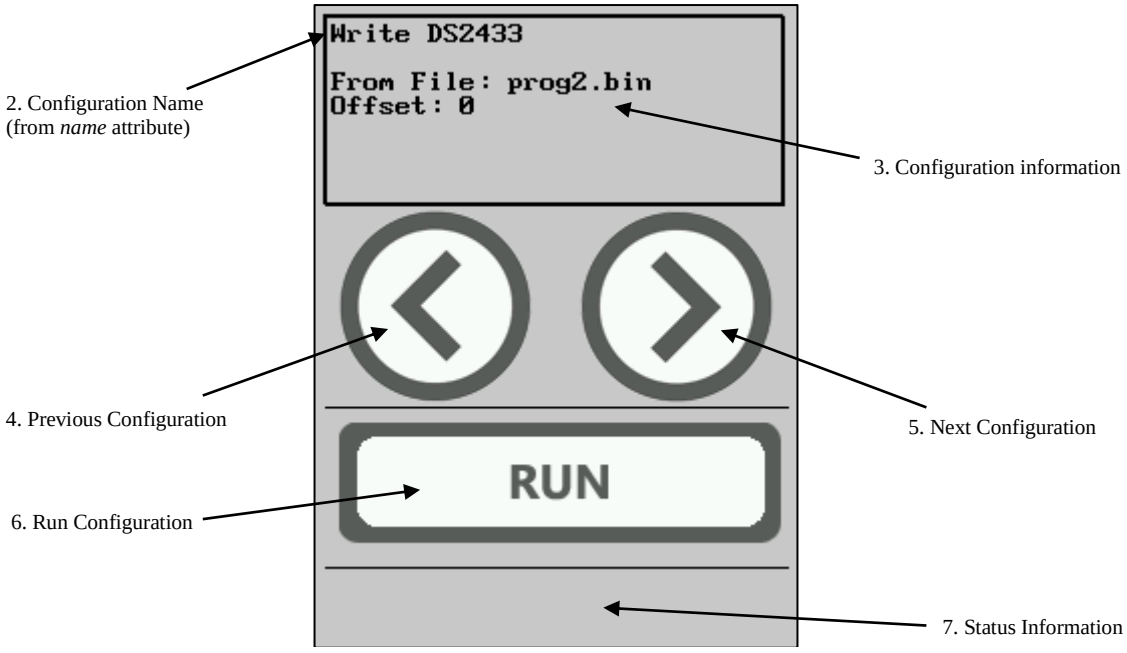
4.2 User Interface

On startup, the following screen will be displayed, which shows the current firmware version.



Startup Screen

Once the software starts and loads the configuration file, the following screen will be displayed.



Configuration Run Screen

1. Firmware Version: Displays the current firmware version number on startup.

2. Configuration Name: Displays the configuration name defined by the user in the configuration file.

3. Configuration Information: Displays the configuration information based on the type of configuration (i.e. Read, Write, etc.). This will show the file name, read offset, data size, etc. as defined by the user in the configuration file.

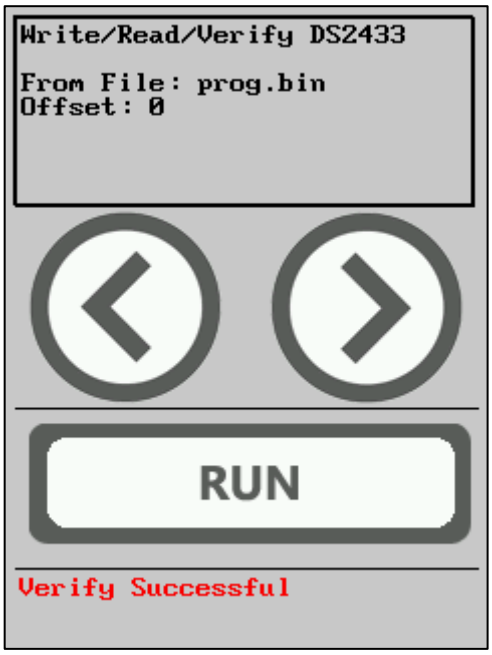
4. Previous Configuration: Button to go to the previous configuration defined in the configuration file. If currently on the first configuration, this will wrap around to the last configuration defined.

5. Next Configuration: Button to go to the next configuration defined in the configuration file. If currently on the last configuration, this will wrap around to the first configuration defined.

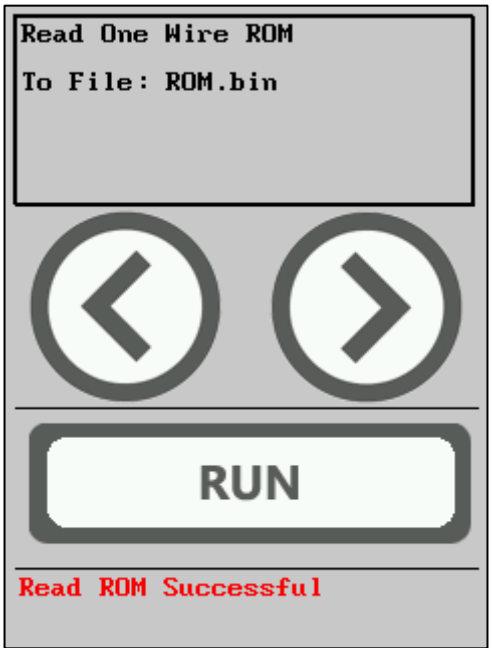
6. Run Configuration: Button to run the currently selected configuration.

7. Status Information: Displays status and error information.

When the Run button is pressed, the displayed configuration will be executed. The status of the execution will be displayed at the bottom of the screen. If there was an error communicating with the 1-Wire device, the result will be displayed in the status information at the bottom of the screen. For example, after running a Write/Read/Verify configuration, the status of each step will be displayed at the bottom of the screen as follows:

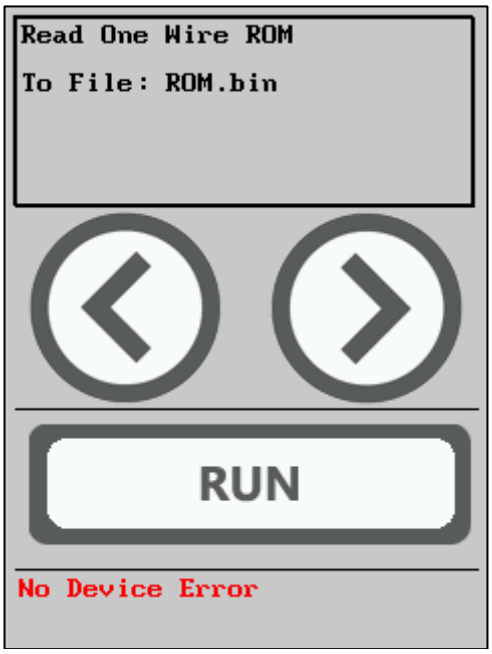


Write/Read/Verify Configuration after Execution



Read ROM Configuration after Execution

If no 1-Wire device is connected and a configuration is run, a “No Device” error will be returned. This is shown below:



4.4 Troubleshooting

The following errors and status messages may be displayed at run time:

Error Message	Possible Cause	Solution
No Device Error	No 1-Wire memory device was detected	Connect a 1-Wire device or check connection
CRC Error	The CRC computed during the read or write operation did not match what was read back from the 1-Wire device or the data location being written is write protected	Ensure the correct 1-Wire device is selected and matches the attached device
Readback Data Error	The data read back from the 1-Wire memory write operation did not match what was written	Ensure the correct 1-Wire device is selected and matches the attached device
No File Found	The specified file to read data from was not found on the SD card	Correct the file name or ensure the file is present
No SD or Config File	No SD card is inserted or the configuration file was not found on the SD card	Insert SD card and ensure config.txt file exists
No Valid License	License has not been loaded	Load license file received from manufacturer onto SD card
Expired License	Current license is not valid for the current release of firmware	Contact support to extend current license or revert firmware

4.5 License File

The license file for your Panther Programmer comes shipped on the SD card. The license file unlocks certain features on you programmer and sets the expiration date for support and firmware upgrades.

The license key is a 25 character alpha-numeric code in the format *AAAAA-BBBBB-CCCCC-DDDDD-EEEEE* and is stored in a file called "license.txt" on the root directory of the SD card. The license file is read from the Programmer on startup. Once the license is read the license is loaded into non-volatile memory and it does not need to remain on the SD card. You only need to place the license file on the SD card if you purchased a new license to either unlock additional features or extend support for additional firmware upgrades.

4.6 Firmware Update Procedure

To update the firmware on the Panther Programmer perform the following steps:

1) Obtain the upgrade file from Aledyne Engineering. The file will be named "panther_V_MAJ_MIN.bin" where MAJ.MIN is the version number.
2) Copy the upgrade file onto the SD card, then insert into the Programmer with the power off.
3) Turn on power by plugging the Programmer into a USB port or USB power adapter.

4) Do not power off the unit during the following step. If power is lost the upgrade will start over once power is re-established (ensure the upgrade file remains on the SD card).
5) An "UPGRADING…" message will appear. This takes up to 1 minute.
6) Then "Upgrading LCD…Please Wait" will appear with status information. This takes about 1 minute to complete.
7) The unit will restart when complete.

If you receive a message stating "Expired License", you will need to contact support@aledyne.com to either revert your firmware or receive a license for extended support in order to download the latest firmware.

4.7 Device Specific Details

Specific devices have certain features that are unique to the family and important considerations are noted here.

1) DS2434/DS2436: These devices have general non-volatile and SRAM sections of memory accessible through a scratchpad. The Panther Programmer will transfer each of these sections to the scratchpad before reading back the commanded address(s) on a read command. On a write command, the Panther Programmer will automatically issue the proper copy command to transfer specified memory segments to non-volatile memory and SRAM from the scratchpad based on the commanded address(s). Also, these devices have capabilities to lock the NV1 section of memory and reset the battery cycle counter. These can be done through the Read/Write Protection commands described in Section 5.0 or they can be read/written through a memory mapped location, which is useful in standalone mode to write/read/verify in a single operation. These memory mapped locations are as follows:

   0x00-0x5F: User accessible memory (refer to datasheet)
   0x60: Writing 0x01 to this address will Lock NV1. Writing 0x00 will Unlock NV1. Reading this address will return the lock/unlock status, where 0x01 is locked and 0x00 is unlocked.

   0x61-0x62: Writing a 0x0000 to this address will clear the battery cycle counter. Reading this address will return the current battery cycle counter where 0x61 is the LSB and 0x62 is the MSB.

5.0 REMOTE INTERFACE

5.0 REMOTE INTERFACE

5.1 Serial Settings

The remote interface is through a virtual COM port over the USB connection via an FTDI FT232 transceiver. The required drivers should be included with Windows or can be downloaded from the FTDI website. The computer serial settings should be set to the following: 19200 Baud, 8 bits, 1 stop bit, no flow control, no parity.

5.2 Packet Structure

5.2.1 Transmit Packet

Data		Size (bytes)	Data
Header	**Start Byte**	1	0x55
	Length	1	5+N
	Command	1	0x00 - 0x06
	Reserved	1	0x00
Body	**Payload**	N	variable
Checksum	**Checksum**	1	sum(start-body)

Start Byte: Start of packet flag (0x55).

Length: The total number of bytes being sent including the Start Byte and Checksum.

Command: The command number.

Reserved: For future use.

Body: Variable length payload for specific command being sent.

Checksum: A simple checksum of all bytes being transmitted. If the sum rolls over, only the LSB of the checksum is used. For example, if the checksum is 0xFF00, the LSB (0x00) will be used for the checksum (modulo checksum method).

5.2.2 Receive Packet

Data		Size (bytes)	Data
Header	**Start Byte**	1	0x55
	Length	1	5+N
	Command Echo	1	0x00 - 0x06
	Command Status	1	variable
Body	**Payload**	N	variable
Checksum	**Checksum**	1	sum(start-body)

Start Byte: Start of packet flag (0x55).

Length: The total number of bytes being sent including the Start Byte and Checksum.

Command Echo: Duplicate of the sent command number.

Command Status:

0x00 – No Error: Command was received properly.

0x01 – In Process: A read or write command is being executed.

0x02 – No Data: No data is available in the read buffer yet.

0x03 – Command Error: The command sent was not recognized.

0x04 – Bad Checksum: The checksum of the received packet was not correct.

0x05 – No License: The serial interface is not licensed. Contact support@aledyne.com.

Body: Variable length payload for specific command response.

Checksum: A simple checksum of all bytes being transmitted. If the sum rolls over, only the LSB of the checksum is used. For example, if the checksum is 0xFF00, the LSB (0x00) will be used for the checksum (modulo checksum method).

5.3 Commands

5.3.1 Get Information

Get Information Packet returns the version number of the Panther Programmer.

Sent:

Start Byte	0x55
Length	0x05
Command	0x00
Reserved	0x00
Checksum	0x5A

Received:

Start Byte	0x55
Length	0x11
Command Echo	0x00
Command Status	0x00
Body 1	Major Ver.
Body 2	Minor Ver.
Body 3-12	Serial Number
Checksum	Checksum

Firmware Version: Major.Minor

Major: Major Version of the firmware

Minor: Minor Version of the firmware

Serial Number: Serial number of programmer (NULL terminated string)

5.3.2 Set Generic Device Properties

Sets the generic 1-Wire device properties. This can be used if the connected 1-Wire memory device is not natively supported. Different parameters can be configured for specific memory devices. Contact support for help configuring properly for your 1-Wire device.

Sent:

Start Byte	0x55
Length	0x0A
Command	0x01
Reserved	0x00
Memory Type	type
CRC Size	size
Address Size	size
Scratchpad Size	size
Scratchpad Delay	delay
Checksum	checksum

Memory Type: The type of memory, either Add-Only or EEPROM, to set for the generic memory device. Add-Only type of devices require a high voltage (12V) programming pulse and typically have a scratchpad size of 1 byte. EEPROM type devices do not require high voltage programming and typically have a scratchpad size greater than 1 byte.

0: Add-Only

1: EEPROM

CRC Size: The size of the CRC generator on the 1-Wire device:

0: No CRC Generator

1: 8-Bit CRC

2: 16-Bit CRC

Address Size: The size of the address sent to the 1-Wire device.

0: Unused

1: 8-Bit Address

2: 16-Bit Address

Scratchpad Size: size in bytes to use for the scratchpad. The scratchpad is a volatile portion of the memory where data is temporarily stored before writing to the non-volatile portion of memory. This should be set to the scratchpad size of the specific memory device or smaller.

Note: Only required for EEPROM Memory Type.

Scratchpad Delay: The delay in milliseconds to wait after issuing a copy scratchpad command, which instructs the 1-Wire device to copy data from the scratchpad to the non-volatile memory

location. Typically this must be at least 10ms but can vary from device to device. **Note: Only required for EEPROM Memory Type.**

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x01
Command Status	0x00
Checksum	0x5B

5.3.3 Get UID

Reads the 8 byte ROM code from the connected one wire memory device.

Sent:

Start Byte	0x55
Length	0x05
Command	0x02
Reserved	0x00
Checksum	0x5C

Received:

Start Byte	0x55
Length	0x0E
Command Echo	0x02
Command Status	0x00
Operation Status	status
UID	8 bytes
Checksum	checksum

Operation Status: Status of the Read ROM operation

0x00 – No Error, 1-Wire device was detected.

0x01 – No 1-Wire device detected.

UID: 1 Byte CRC code, 6 byte Serial Number, then 1 byte Family Code. For example, for the DS2502:

8–Bit CRC Code		48–Bit Serial Number		8–Bit Family Code (09h)	
MSB	LSB	MSB	LSB	MSB	LSB

5.3.4 Read Data

Reads data from the specified 1-Wire memory device at a given offset. This command initiates the read and responds immediately if the command was accepted. The read operation will occur in the background and the *Read Buffer* command must be used to check the status of the read operation and retrieve the actual data from the read.

Sent:

Start Byte	0x55
Length	0x09
Command	0x03
Reserved	0x00
Offset (LSB)	variable
Offset (MSB)	variable
Size	0x00-0x80
Device	variable
Checksum	checksum

Offset: The offset to start the read from (little endian – LSB, MSB).

Size: The number of bytes to read. To read more than 128 bytes, subsequent commands must be issued with the appropriate offset.

Device: The 1-Wire device type or Family Code.

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x03
Command Status	status
Checksum	checksum

Command Status:

0x00 – No Error: Command was received properly and read operation will commence.

0x01 – In Process: A read or write command is already being executed. Wait for the previous operation to complete.

5.3.5 Write Data

Writes data to the specified 1-Wire memory device at a given offset. This command stores the data into an internal memory buffer then responds immediately if the command was accepted. The write operation will occur in the background and the *Read Buffer* command must be used to check the status of the write operation.

Sent:

Start Byte	0x55
Length	0x09+N bytes
Command	0x04
Reserved	0x00
Offset (LSB)	variable
Offset (MSB)	variable
Size	0x00-0x80
Device	variable
Data (128 bytes max)	N bytes
Checksum	checksum

Offset: The offset to start the write to (little endian – LSB, MSB).

Size: The number of bytes to write. To write more than 128 bytes, subsequent commands must be issued with the appropriate offset.

Data: The data to write to the 1-Wire device at the given offset. This can be at most 128 bytes. To write more than 128 bytes, subsequent commands must be issued with the appropriate offset.

Device: The 1-Wire device type or Family Code.

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x04
Command Status	status
Checksum	checksum

Command Status:

0x00 – No Error: Command was received properly and write operation will commence.

0x01 – In Process: A read or write command is already being executed. Wait for the previous operation to complete.

5.3.6 Read Buffer

Checks the status of a read or write operation. If a read operation has bene complete, the response will return the requested data read from the attached 1-Wire device.

Sent:

Start Byte	0x55
Length	0x05
Command	0x05
Reserved	0x00
Checksum	0x5F

Received:

Start Byte	0x55
Length	0x06+N bytes
Command Echo	0x05
Command Status	status
Operation Status*	status
Size*	0x00-0x80
Buffer Data (128 bytes max)*	N bytes
Checksum	checksum

**Only present when write/read command is complete (Command Status not In Process)*

Command Status:

0x00 – No Error: Command was received properly and response contains requested data.

0x01 – In Process: A read or write command is in process. Continue to call *Read Buffer* until not In Process.

0x02 – No Data: No data is available in the read buffer. This will be the case for a write operation.

Operation Status: Status of the Write/Read operation

0x00 – No Error, 1-Wire device was detected.

0x01 – No 1-Wire device detected.

0x02 – CRC error during write/read operation.

0x03 – Readback data error during write operation.

0x04 – File access error.

0x05 – Write command was not successful.

Size: The number of bytes returned from the buffer.

Buffer Data: The data from the buffer of a Read operation. This can be at most 128 bytes.

5.3.7 Execute Configuration

Runs a configuration loaded from the SD card of the programmer. If the selected configuration is out of range, the last configuration will be executed.

Sent:

Start Byte	0x55
Length	0x06
Command	0x06
Reserved	0x00
Config Number	variable
Checksum	checksum

Config Number: The SD card configuration number to execute. If this is out of range, the last configuration will be executed.

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x03
Command Status	status
Checksum	checksum

5.3.8 Read Protection

Reads the block protection for the block that corresponds to the provided address. To read the protection status of a single block, set the Size to 1. To read the protection status of multiple blocks, the Size can be set to a number greater than 1. If the Size is 4, then the block protection for the 4 blocks of data starting from the Address provided will be returned. If the block is unprotected, the code is 0x0F, and if the block is protected, the code is 0xF0. This command specifically applies to the DS28E80, which enables block protection features. This command is also used to read specific registers on the DS2434/36 by specifying a starting register number in the address field. The read operation will occur in the background and the *Read Buffer* command must be used to check the status of the read operation and retrieve the block protection data from the read.

Sent:

Start Byte	0x55
Length	0x09
Command	0x07
Reserved	0x00
Address (LSB)	variable
Address (MSB)	variable
Size	0x00-0x80
Device	variable

Checksum	checksum

Address: The address of the block to read the block protection status (little endian – LSB, MSB). The address can be any address in the block range. Or the starting address of register to read from a DS2434/36.

Size: The number of block protection status bytes to read or register bytes to read. To read more than 128 bytes, subsequent commands must be issued with the appropriate offset.

Device: The 1-Wire device type or Family Code.

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x07
Command Status	status
Checksum	checksum

Command Status:

0x00 – No Error: Command was received properly and read operation will commence.

0x01 – In Process: A read or write command is already being executed. Wait for the previous operation to complete.

5.3.9 Write Protection

Used to protect the user memory block that corresponds to the provided address from changes. Once set, the protection cannot be reset. The write operation will occur in the background and the *Read Buffer* command must be used to check the status of the protect operation. This command is also used to send specific commands to the DS2434/36 by specifying the command number in the address field.

Sent:

Start Byte	0x55
Length	0x09
Command	0x08
Reserved	0x00
Address (LSB)	variable
Address (MSB)	variable
Size	0x00
Device	variable
Checksum	checksum

Address: The address of the block to protect (little endian – LSB, MSB). The address can be any address in the block range. Or the command to execute on a DS2434/36.

Size: Unused (0x00)

Device: The 1-Wire device type or Family Code.

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x08
Command Status	status
Checksum	checksum

Command Status:

0x00 – No Error: Command was received properly and protect operation will commence.

0x01 – In Process: A read or write command is already being executed. Wait for the previous operation to complete.

5.3.10 Read Remaining Cycles

Reads how many more times a Write Block command can be executed for the memory block that corresponds to the provided address. To read the remaining cycles of a single block, set the Size to 1. To read the remaining cycles of multiple blocks, the Size can be set to a number greater than 1. If the Size is 4, then the remaining cycles for the 4 blocks of data starting from the Address provided will be returned. The value for an unprogrammed memory block is 0x08. The value 0x00 indicates that the write cycles for the block are exhausted. This command specifically applies to the DS28E80, which limits the number of write cycles. The read operation will occur in the background and the *Read Buffer* command must be used to check the status of the read operation and retrieve the remaining cycles data from the read.

Sent:

Start Byte	0x55
Length	0x09
Command	0x09
Reserved	0x00
Address (LSB)	variable
Address (MSB)	variable
Size	0x00-0x80
Device	variable
Checksum	checksum

Address: The address of the block to read the remaining cycles (little endian – LSB, MSB). The address can be any address in the block range.

Size: The number of block remaining cycle bytes to read. To read more than 128 bytes, subsequent commands must be issued with the appropriate offset.

Device: The 1-Wire device type or Family Code.

Received:

Start Byte	0x55
Length	0x05
Command Echo	0x09
Command Status	status
Checksum	checksum

Command Status:

0x00 – No Error: Command was received properly and read operation will commence.

0x01 – In Process: A read or write command is already being executed. Wait for the previous operation to complete.

5.4 Pseudo-Code Examples

To write or read from an attached 1-Wire memory, multiple write/read and read buffer commands must be sent if the data size writing or reading is greater than 128 bytes. This is because the data buffer on the Panther Programmer is limited to 128 bytes to keep serial communication concise. The following examples show the general sequence for writing and reading data using the command set above. This is only required if you are designing a PC application that is not based on LabVIEW. If you are using LabVIEW, a free driver is available for download on the LabVIEW Tools Network.

5.4.1 Write Data

Assume DS2433 (4Kb or 512 bytes), Writing 200 bytes (value of 0x00) starting at offset 100.

1) Break out first 128 bytes of data to write.

2) Send Write Data (0x04) packet for first 128 bytes with device code 35 (0x23), offset of 100 (0x0064), size of 128 (0x80), and data.
   Packet (Hex): 5589 0400 6400 8023 0000 0000 …. 0000 E9
   Response (Hex): 5505 0400 5E

3) Send Read Buffer (0x05) packet until Command Status is not In Process (0x01).
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5505 0501 60 (In process)
   …
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5507 0502 0000 63 (Done – No Data)

4) Break out remaining 72 bytes of data to write.

5) Send Write Data (0x04) packet for next 72 bytes with device code 35 (0x23), offset of 228 (0x00E4), size of 72 (0x48), and data.
   Packet (Hex): 5551 0400 E400 4823 0000 0000…. 0000 F9
   Response (Hex): 5505 0400 5E

6) Send Read Buffer (0x05) packet until Command Status is not In Process (0x01).
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5505 0501 60 (In process)
   …
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5507 0502 0000 63 (Done – No Data)

5.4.2 Read Data

Assume DS2433 (4Kb or 512 bytes), Reading 200 bytes starting at offset 100.

1) Send Read Data (0x03) packet for first 128 bytes with device code 35 (0x23), offset of 100 (0x0064), and size of 128 (0x80).
   Packet (Hex): 5509 0300 6400 8023 68
   Response (Hex): 5505 0300 5D

2) Send Read Buffer (0x05) packet until Command Status is not In Process (0x01). Data is returned in Read Buffer packets when status indicates it is complete.
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5505 0501 60 (In process)
   …
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5587 0500 0080 **0000 0000 …. 0000** 61 (Done – With Data)
   *****Bold*** *is returned data*

3) Send Read Data (0x03) packet for next 72 bytes with device code 35 (0x23), offset of 228 (0x00E4), and size of 72 (0x48).
   Packet (Hex): 5509 0300 E400 4823 B0
   Response (Hex): 5505 0300 5D

4) Send Read Buffer (0x05) packet until Command Status is not In Process (0x01). Data is returned in Read Buffer packets when status indicates it is complete.
   Packet (Hex): 5505 0500 5F
   Response (Hex): 5505 0501 60 (In process)
   …
   Packet (Hex): 5505 0500 5F
   Response (Hex): 554F 0500 0048 **0000 0000 …. 0000** F1 (Done – With Data)
   *****Bold*** *is returned data*

6.0 LABVIEW DRIVER

6.0 LABVIEW DRIVER

6.1 Description

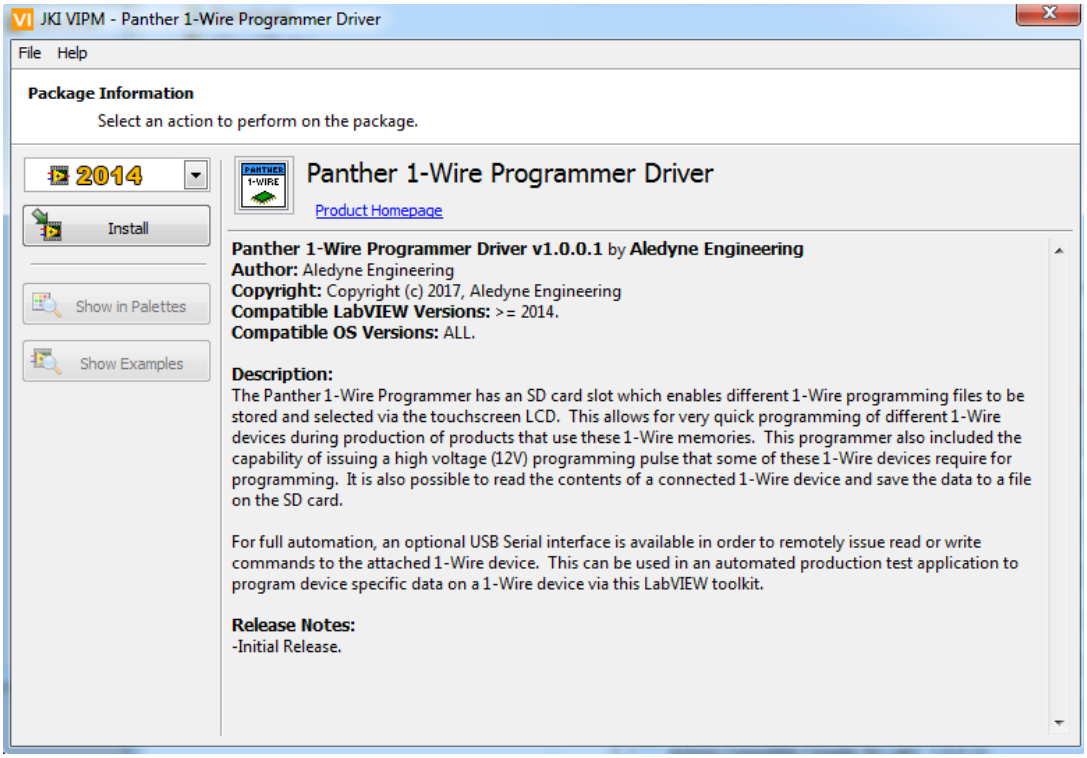
A LabVIEW plug and play driver that has been certified by National Instruments is available for free on the National Instruments LabVIEW Tools Network. The driver can be installed through VI Package Manager (VIPM) and is compatible with LabVIEW 2014 and later. The driver implements all the required functions as described in the Remote Interface chapter in order to read and write to the attached 1-Wire device. This is ideal when used in a production test fixture and multiple units need to be programmed in production. The LabVIEW driver allows one to integrate the Panther Programmer with a production test fixture using TestStand and/or LabVIEW in order to program unique data onto the attached 1-Wire memory device. The auxiliary 1 Wire connector can be used to permanently attached to a custom bed of nails or smart cable test fixture.

6.2 Requirements

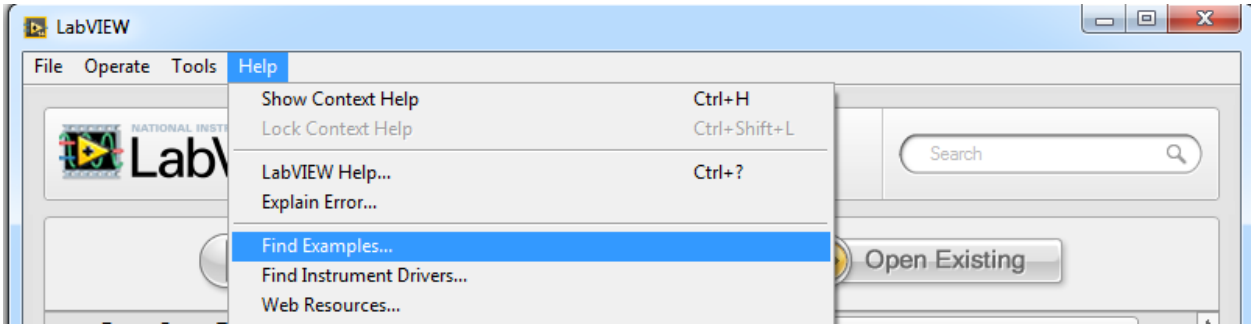
- LabVIEW 2014 or higher
- NI-VISA 14.0 or higher

6.3 Installation

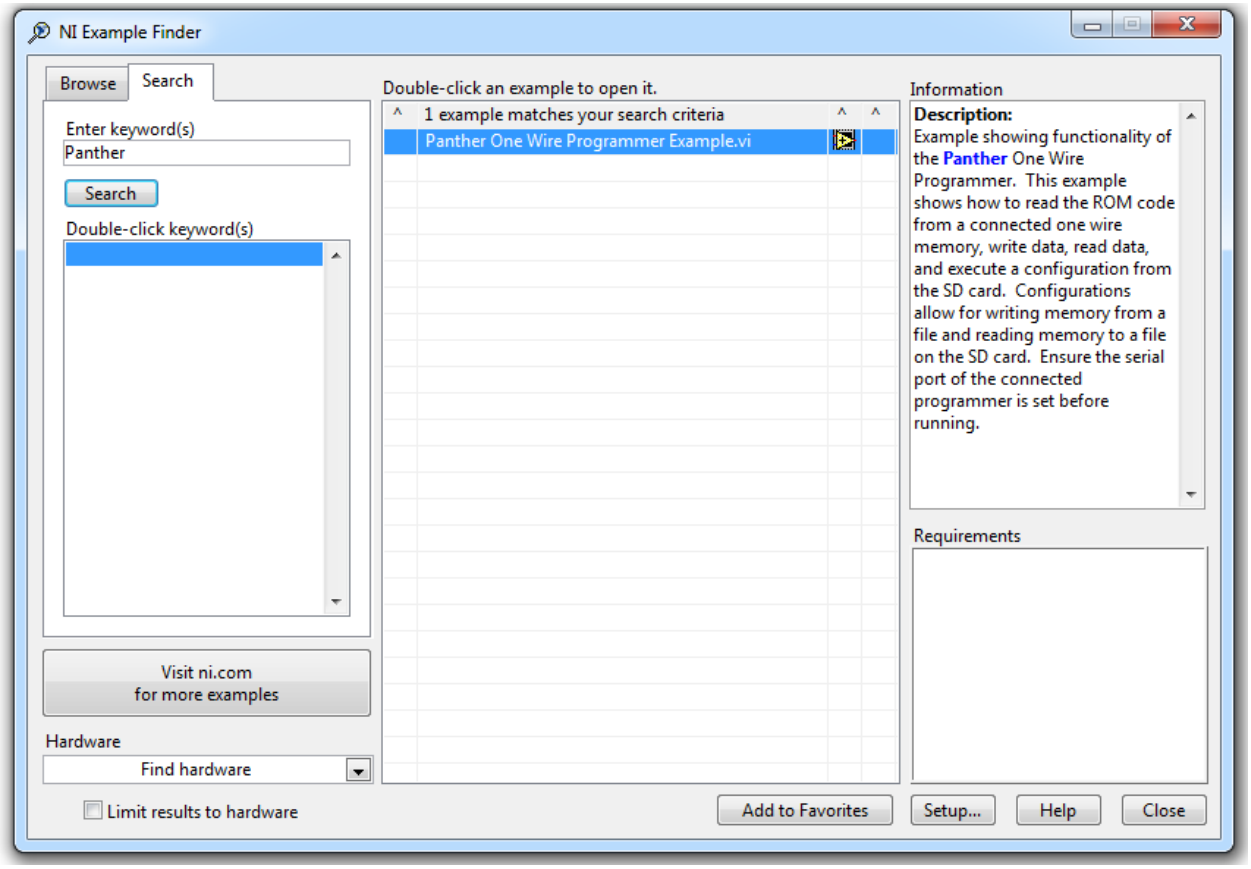
Open VI Package Manager (VIPM) and type “Panther” in the search field. Then select the Panther Programmer 1-Wire Driver and select “Install”.



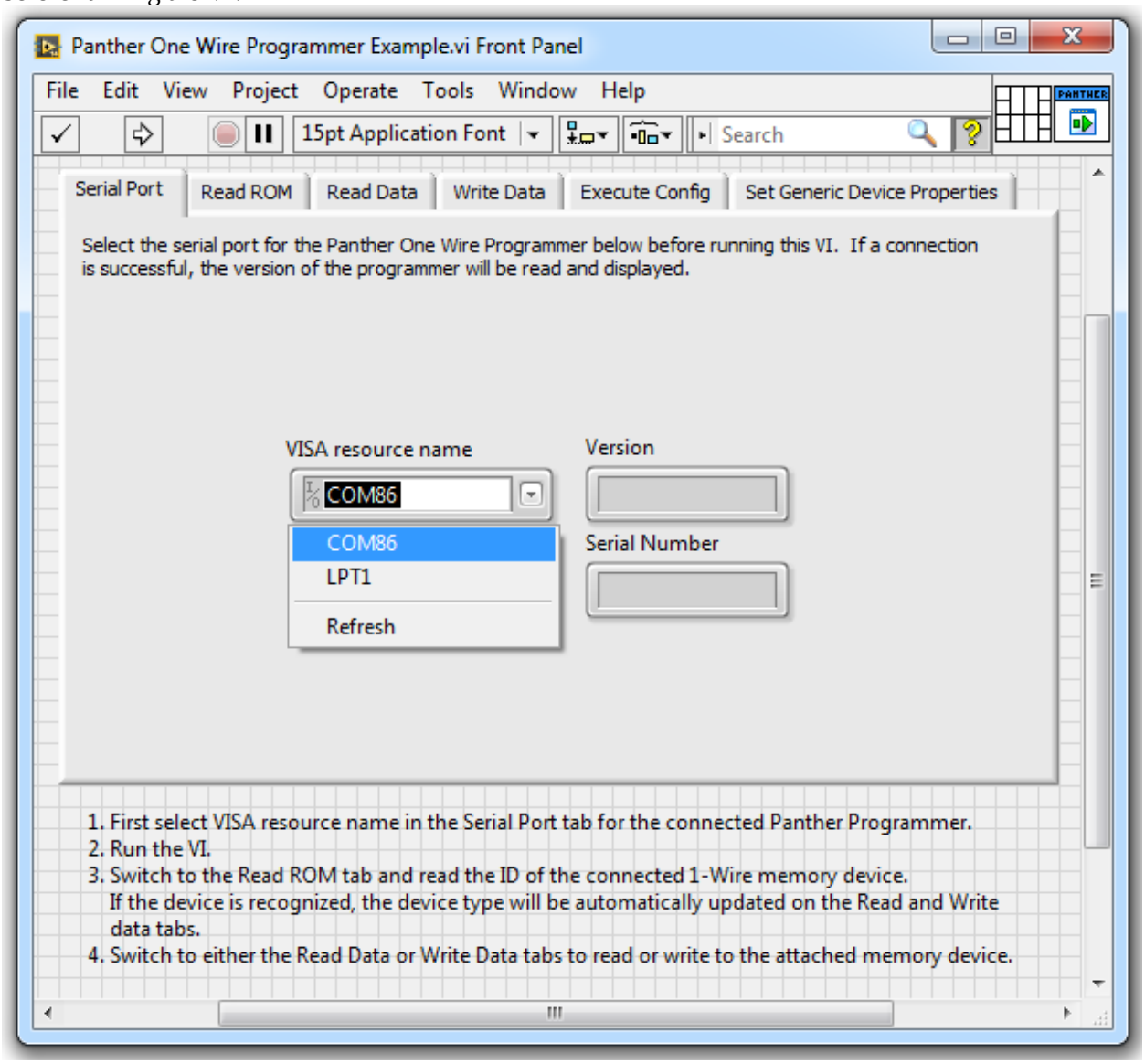
Once installed, a ready to use example shows how to use the driver. To find the example, open the Example Finder from *Help>Find Examples…*



Then type in "Panther" in the search box and press *Search*.



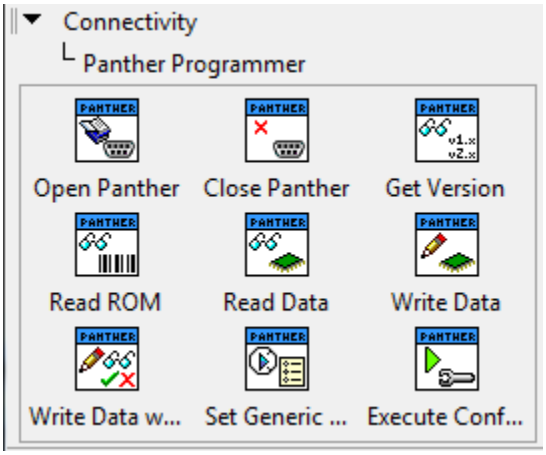
Double-click on the Panther One Wire Programmer Example.vi. Once the example opens, make sure to select the COM port of the connected Panther programmer from the VISA resource name drop down before running the VI.



Once the COM port is selected, run the VI and the Version will be returned if the connection was successful. You can now use the tabs to perform different functions like Read the 8-byte ROM code, Read Data, Write Data, etc.

6.4 LabVIEW Palette

This Panther Programmer functions palette is located under the Connectivity palette. The palette is organized as follows:



6.5 Palette VIs

6.5.1 Open Panther

Panther One Wire Programmer.lvlib:Open Panther.vi
Open connection to Panther One Wire Programmer. Also requests the version of the hardware.

Connector Pane:

Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information

about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

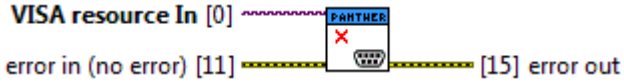
Version is the version returned from the programmer.

Serial Number is the serial number returned from the programmer.

6.5.2 Close Panther

Panther One Wire Programmer.lvlib:Close Panther.vi
Close connection to Panther One Wire Programmer.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

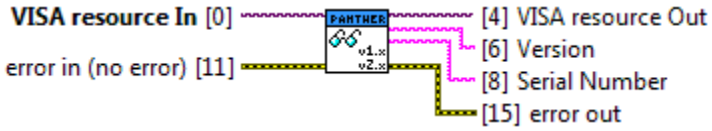
source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

6.5.3 Get Version

Panther One Wire Programmer.lvlib:Get Version.vi
Returns the version and serial number of the Panther One Wire Programmer hardware.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

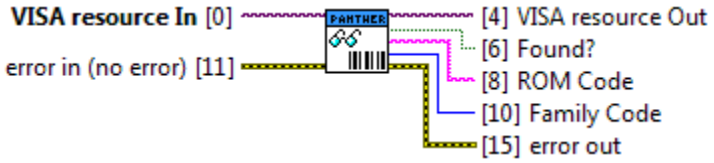
Version is the version returned from the programmer.

Serial Number is the serial number returned from the programmer.

6.5.4 Read ROM

Panther One Wire Programmer.lvlib:Read ROM.vi
Reads the 8 byte ROM code from the connected one wire memory device.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

ROM Code is the 8 byte ROM read from the connected memory device.

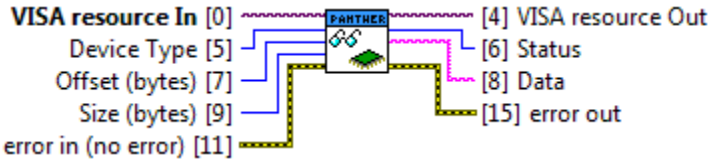
Found? is TRUE if a memory device is present, FALSE if not present.

Family Code is the family code detected from the device.

6.5.5 Read Data

Panther One Wire Programmer.lvlib:Read Data.vi
Reads data from the specified one wire memory device at a given offset. If a device is not listed, generic properties can be set using Set Generic Device Properties.vi, then the Generic device type can be selected.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

Offset (bytes) is the byte offset to read from.

Size (bytes) is the number of bytes to read.

Device Type is the one wire memory device type.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

Data is the data read.

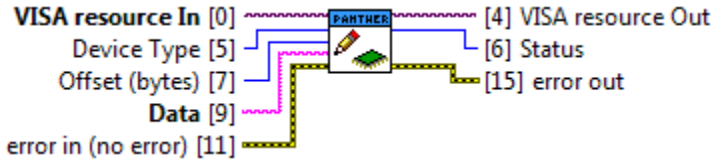
Status is the status returned from the programmer.

6.5.6 Write Data

Panther One Wire Programmer.lvlib:Write Data.vi

Writes data to the specified one wire memory device at a given offset. If a device is not listed, generic properties can be set using Set Generic Device Properties.vi, then the Generic device type can be selected.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

Data is the data to write.

Offset (bytes) is the byte offset to read from.

Device Type is the one wire memory device type.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE

(checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

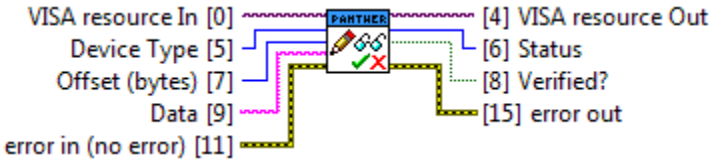
Status is the status returned from the programmer.

6.5.7 Write Data with Verify

Panther One Wire Programmer.lvlib:Write Data with Verify.vi

Writes data to the specified one wire memory device at a given offset, then reads the same data size back from the same offset and confirms if the read back data matches the written data.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

Data is the data to write.

Offset (bytes) is the byte offset to read from.

Device Type is the one wire memory device type.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE

(checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

Verified? is TRUE if read back data matches written data.

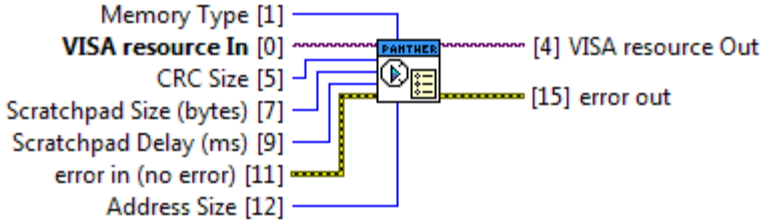
Status is the status returned from the programmer.

6.5.8 Set Generic Device Properties

Panther One Wire Programmer.lvlib:Set Generic Device Properties.vi

If a device is not listed in the compatible device types, a generic device can be configured. The generic properties can be set with this vi and then the Generic device type can be used for read and write operations.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

Memory Type is the type of memory, Add-Only or EEPROM, to set for the generic memory device.

CRC Size sets the CRC size to use for the generic memory device.

Scratchpad Size (bytes) sets the size of the scratchpad for the generic memory device used for memory writes.

Scratchpad Delay (ms) sets the delay in ms after a scratchpad copy for the generic memory device.

Address Size sets the address size to use for the generic memory device.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

6.5.9 Execute Config

Panther One Wire Programmer.lvlib:Execute Config.vi
Runs a configuration loaded from the SD card of the programmer. If the selected configuration is out of range, the last configuration will be executed.

Connector Pane

Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

Config Number is the configuration number on the SD card.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

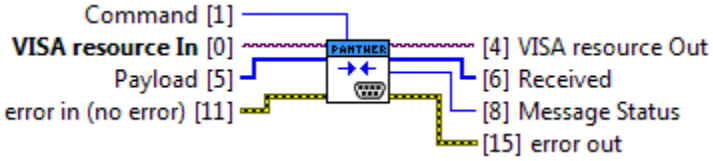
The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out is the duplicate Serial port.

6.5.10 Send Panther Command

Panther One Wire Programmer.lvlib:Send Panther Command.vi
Sends the specified serial packet to the programmer and waits for a reply. If an error is detected, a specific error type is generated.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In is the input Serial port.

Payload is the data to write as the payload part of the packet (not including the header and checksum).

Command is the command type for the programmer.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE

(checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out VISA resource Out is the duplicate Serial port.

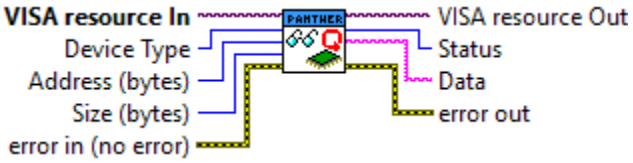
Received is the payload data read back (not including the header and checksum).

Message Status is the status returned from the programmer.

6.5.11 Read Remaining Cycles

Panther One Wire Programmer.lvlib:Read Remaining Cycles.vi
Reads the remaining write cycles of the requested block based on the input address to the specified one wire memory device. If **Size** is greater than 1, the remaining cycles for the number of blocks specified will be returned.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In VISA resource In is the input Serial port.

Address (bytes) Address (bytes) is the address of the block to read remaining cycles of.

Size (bytes) Size (bytes) is the number of bytes to read, or the number of blocks to read the remaining cycles of.

Device Type Device Type is the one wire memory device type.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about

the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out VISA resource Out is the duplicate Serial port.

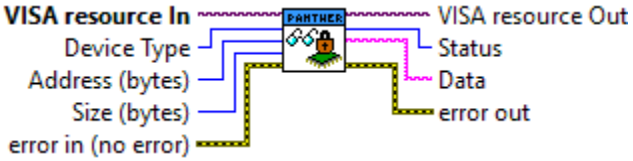
Data Data is the data read.

Status Status is the status returned from the programmer.

6.5.12 Read Protection

Panther One Wire Programmer.lvlib:Read Protection.vi
Reads the protection state of the requested block based on the input address to the specified one wire memory device. If **Size** is greater than 1, the protection state for the number of blocks specified will be returned. The connected one wire memory must support block protection.

Connector Pane



Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In VISA resource In is the input Serial port.

Address (bytes) Address (bytes) is the address of the block to read the protection status of.

Size (bytes) Size (bytes) is the number of bytes to read, or the number of blocks to read the protection status of.

Device Type Device Type is the one wire memory device type.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out VISA resource Out is the duplicate Serial port.

Data Data is the data read.

Status Status is the status returned from the programmer.

6.5.13 Write Protection

Panther One Wire Programmer.lvlib:Write Protection.vi
Write protects the requested block based on the input address to the specified one wire memory device. The connected one wire memory must support block protection.

Connector Pane

Controls and Indicators

error in (no error) The **error in** cluster can accept error information wired from VIs previously called. Use this information to decide if any functionality should be bypassed in the event of errors from other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource In VISA resource In is the input Serial port.

Offset (bytes) Offset (bytes) is the byte offset to read from.

Device Type Device Type is the one wire memory device type.

error out The **error out** cluster passes error or warning information out of a VI to be used by other VIs.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

status The **status** boolean is either TRUE (X) for an error, or FALSE (checkmark) for no error or a warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

code The **code** input identifies the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

source The **source** string describes the origin of the error or warning.

The pop-up option **Explain Error** (or Explain Warning) gives more information about the error displayed.

VISA resource Out VISA resource Out is the duplicate Serial port.

Status Status is the status returned from the programmer.

7.0 SPECIFICATIONS

7.0 SPECIFICATIONS

Specification	Description
Electrical:	
Supply Voltage	5 VDC
Supply Current	< 200 mA
Physical:	
Dimensions	150 mm x 92 mm x 28 mm
Weight	175 grams
Operating Temperature	5˚C – 50˚C
Storage Temperature	-20˚C – 80˚C
Relative Humidity	<90%
I/O Ports:	
Serial Interface/Power	FTDI-232 USB
TO-92 1-Wire Connector	Data Pin: ESD Protection exceeds 15kV
SO 1-Wire Connector	Data Pin: ESD Protection exceeds 15kV
Auxiliary 1-Wire Connector	Data Pin: ESD Protection exceeds 15kV
SD Card Interface	3.3V FAT16 or FAT32
1-Wire Communication:	
Communication Voltage	3.3V
Add-Only Programming Voltage	12V

8.0 PACKAGE CONTENTS

8.0 PACKAGE CONTENTS

The following items are included in the package:

- Panther 1-Wire Programmer
- Micro USB Cable, 2m
- 256MB+ SD Card
- OPTIONAL – SOP Adapter (150MIL or 208MIL), TDFN Adapter

Revision Sheet

Release No.	Date	Revision Description
Rev. 1.00	10/13/17	User's Manual Created
Rev. 1.01	11/12/17	Added serial number query to remote interface and LabVIEW driver
Rev. 1.02	04/15/19	Version 1.2 Firmware: Added append and ASCII modes to ROM read, added overdrive support
Rev. 1.03	02/19/20	Version 1.4 Firmware: Added PIO control and DS28E05 updated timings and cable length note.
Rev. 1.04	08/28/23	Version 2.1 Firmware: Support for DS28E80 and block protection. OEM board support.
Rev. 1.05	02/20/24	Version 2.2 Firmware: Support for DS2434, DS2436, DS28EC20, DS28E04, and Memory iButtons (DS1971, DS1973, DS1992, DS1993, DS1995, DS1996).