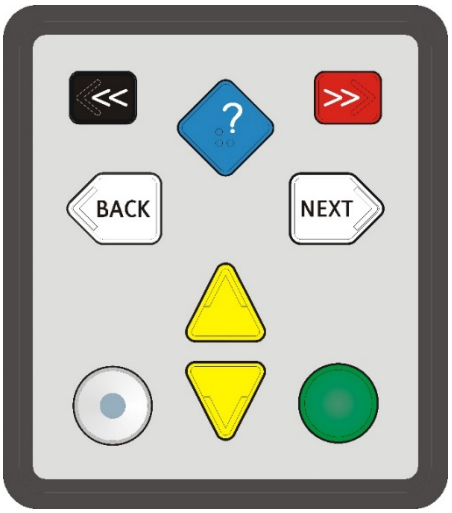


API for controlling the Keypad from the Host Computer

This details how the NavPad™ can be controlled from a host that has USB capabilities.

The API incorporating this command set is downloadable from *www.storm-interface.com.*

Host API Library - Overview

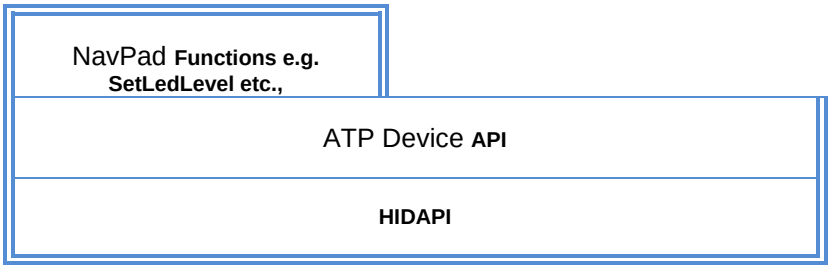
The Host API Library is a middleware application between the host application and the NavPad. This is available as a download together with the HIDAPI library.

- The API library allows for the host application to invoke the NavPad functions as listed.
  It encapsulates all the communications to USB and provides a simple API for the host application developers.

- HIDAPI - This is a third party library, which allows an application to interface with USB HID-Compliant devices on Windows, Linux, and Mac OS X. While it can be used to communicate with standard HID devices like keyboards, mice, and Joysticks, it is most useful with custom (Vendor-Defined) HID devices. This allows for host software to scan for the device using its VID/PID.

The NavPad uses USB for communicating with the host. It includes an HID-compliant device One advantage of using this implementation, which uses only HID interfaces, is that no drivers are required on the host system.

The protocol for communicating with host is described fully in the following pages. The basic architecture is shown below.



With this approach the developer does not need to worry about the communication at low level.

The library can be ported to your specific platform if required.

Currently the library has been tested on Windows and Linux (Ubuntu) platform.

Device Communications and Message Format

NAVPAD™ keypad uses the ASCII/binary Message format described below. Every message that is sent from a host should be acknowledged with the control byte ACK (0x06). A retransmission should be initiated if an NAK (0x15) is received or if no acknowledge is received at all.

Message Formats

A	Alpha character, 'A'-'Z' and 'a' - 'z'
C	Control character one byte in length.
H	Hexadecimal characters, ‘0’-‘9’, ‘A’-‘F’
N	Numeric character, ‘0’-‘9’
S	Special characters, entire character set 0x00 - 0xFF

ASCII Message Format

	Message Field	Type	Length	Description
1	STX	C	1	Control character Start of Text = 0x02
2	Message Id	H	2	Defines the type of message and format of the data field
3	Data Length	H	2	Hexadecimal value represented in ASCII defines the number of bytes in the data field. '00' to 'FF'. Maximum data field size is 256 bytes.
4	Data Field	S	var	In binary format
5	ETX	C	1	Control character ETX = 0x03
6	LRC	C	1	Longitudinal Redundancy Check Digit, calculated on all previous data including STX

Controlling the Keypad from the Host Computer

Message Definitions and Error Codes

Here is a general table describing the message Ids, more detailed descriptions for each message Id follows. When a message is one way only, the Message Id. is the same for both the message and response.

ID.	Message	Description
01	Device Status Request	Host To NAVPAD™ keypad – Output the firmware version and all currently selected parameters
02	LED Brightness	Host To NAVPAD™ keypad – adjust key led brightness. (default: 6)
03	Jack LED	Host To NAVPAD™ keypad – adjust Jack led brightness (default 6)
04	Key Press Beeper On/Off	Host To NAVPAD™ keypad - Enable/Disable beeper. (Default: Enable)
05	Load New code table	Host To NAVPAD™ keypad – Load new code table
06	Beeper Command	Host To NAVPAD™ keypad – sound beeper
07	Keypad Table	Host To NAVPAD™ keypad – Select layout table 0 – Default Table 1 – Alternate Table 2 – Customised
08	Reserved	Reserved
09	Write to default	Host To NAVPAD™ – NavPad™ writes configuration data from ram to flash.
10	Reset to factory default	Host To NAVPAD™ – Reset device back to factory default
11	RESERVED	Reserved
12	Enable BSL	Host To NAVPAD™ – Sets the NavPad™ to detect the device loader for firmware loading
13	Status Beeper	Host to NAVPAD™ – Sounds the beeper for x period. X is passed in value (0 - 9). This will sound the beeper even if beeper is switched off.

Error Code

Every response message contains one of the following error codes:

00	No error
01	Command not recognized
02	Command not support at this stage
03	Parameter not supported
04	Hardware fault

.

Controlling the Keypad from the Host Computer

List of Messages

(Structure of Messages from Host to NavPad™ is on the following pages)

ID	Name	Description
01	Device Status Request	Output the firmware version & selected parameters

Brightness functions are dependent on which product is being controlled

		Non-illuminated NavPad	*Illuminated NavPad*
02	LED Brightness	Adjust Jack LED Brightness	Adjust Key LED brightness
03	Jack LED Brightness		Adjust Jack LED brightness

04	Key Press Buzzer	Enable/Disable buzzer.
05	Load New code table	Load new code table
06	Buzzer Command	Sound buzzer
07	Keypad Type	Select layout code table
08	Reserved	
09	Write to default	NavPad™ writes configuration data from ram to flash
10	Reset to factory default	Reset device back to factory default
11	Enable BSL	Sets the NavPad™ to detect the device loader for firmware loading
13	Status Buzzer	Sounds the buzzer for x period.

Structure of Messages from NavPad™ to Host

01 Key Press Code	sends a key scan code back to HOST when a key is pressed on keypad

Device Status (01)

Host sends this message to request the status of the NAVPAD™ keypad

NAVPAD™ **Status Response**

Secure device sends this message to Host in response to the Device Status message.

	Data Field	Type	Length	Description
ec	Error Code	SH	2	
Lb	LED Brightness	SN	1	Value (0 – 9)
Jl	Jack LED Brightness	SN	1	Value (0 – 9)
Bz	Beeper	SN	1	0 – OFF, 1 – ON
Ky	Keypad type	SN	1	0 – 5 way, 1 – 6 way, 2 – 8 way
Bp	Beeper Period	SN	1	0 – 9
Kt	Keypad Table	SN	1	0 – Default Table 1 – Alternate Table 2 – Customised Table
Kc	Keycode	SH	20	Customised keycode for each key
fw	Firmware Version	ANS	20	Left justified, if Firmware Version is less than 20 then just add enough spaces after the Firmware Version until this field is completed, for instance, “123456” becomes: “123456 “

Host sends this message to request information from the NAVPAD™

Host Device		NAVPAD™
[01]	⇒	
	⇐	[01][ec][lb[lc][bz][kt][bp][kt][20*kc][fw]

LED Brightness Command (02)

Host sends this message to control brightness of LEDs

	Data Field	Type	Length	Description
1	LED brightness	SN	1	0 - 9

LED Brightness Command Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NAVPAD™
[02][lb]	⇨	
	⇦	[02][ec]

Note: LED brightness of 0 value indicates LEDs are off

LED brightness of 9 value indicates full brightness

Jack LED Brightness Command (03)

Host sends this message to control brightness of Jack LEDs

	Data Field	Type	Length	Description
1	Jack LED brightness	SN	1	0 - 9

Jack LED Brightness Command Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NAVPAD™
[02][lb]	⇨	
	⇦	[02][ec]

Note: Jack LED brightness of 0 value indicates LEDs are off

Jack LED brightness of 9 value indicates full brightness

Key Press Beeper Command (04)

Host sends this message to enable/disable beeper on key presses

	Data Field	Type	Length	Description
1	Beeper	SN	1	0-Disable, 1-Enable

Beeper Command Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NavPad™
[04][bp]	⇨	
	⇦	[04][ec]

Load New Key Code Table Command (05)

Host sends this message to Load New Code Table

	Data Field	Type	Length	Description
1	Load New Code Table	SH	20	Key Code Table:

Load New Table Command & Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NavPad™
[05][lt][32 scan codes]	⇒	
	⇐	[05][ec]

Note: Length is always 20,

Format of table is as follows:

<modifier for key 1><code for Key 1><modifier for key 2><Code for Key 2>.....etc

The code table is specified in the user manual together with the modifier code. For example to program the following for 4 way :

Key 1 – A

Key 2 – a

Key 3 – 9

Key 4 - !

<0xE1><0x04><0x00><0x04><0x00><0x26><0xE5><0x1E>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>< 0x00><0x00>

Note: 20 bytes must be sent, for unused key code pad the values with 0x00.

Note: For shift modifiers there is a left and right modifiers value defined. So we can use 0xE1 – Left Shift and 0xE5 – Right shift. Similarly there is left and right Alt

Buzzer Command (06)

Host sends this message to start buzzer for specified duration

	Data Field	Type	Length	Description
1	Duration	SN	1	Value 0 - 9

Beeper Command &Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NavPad™
[06][bp]	⇒	
	⇐	[06][ec]

Buzzer duration value increments in 0.25s. For example 1 = 0.25s, 2 – 0.5s, 3 – 0.75s, 4 – 1.0s etc.,

Keypad Table Command (07)

Host sends this message to set code table to be used.

	Data Field	Type	Length	Description	
1	Code Table	SN	1	0 – Default Table 1 – Alternate Table 2 – Customised Table	

Keypad Command & Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NavPad™
[07][bp]	⇨	
	⇦	[07][ec]

Reserved (08)

Write Config Data To Flash command (09)

Host sends this command to request the NAVPAD™ to write the configuration data from RAM to FLASH.

This command has no data associated with it.

RAM to FLASH Command & Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NAVPAD™-VOIP
[09]	⇒	
	⇐	[09][ec]

Reset To Factory Default command (10)

Host sends this command to request the NAVPAD™ to reset parameters back to factory default.

This command has no data associated with it.

Reset To Factory Default Command & **Response**

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NAVPAD™
[10]	⇒	
	⇐	[10][ec]

Enable BSL Command (12)

Host sends this command to request the NAVPAD™ to start downloader

Enable BSL **Command & Response**

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NAVPAD™
[12]	⇨	
	⇦	[12][ec]

Status Buzzer command (13)

Host sends this message to start buzzer for specified duration

	Data Field	Type	Length	Description
1	Duration	SN	1	Value 0 - 9

Status Beeper Command & Response

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		NAVPAD™
[13][sb]	⇒	
	⇐	[13][ec]

Key Press Code

With the USB stack configured for a standard keyboard interface, the NavPad™ sends appropriate key report to HOST when a key is pressed on keypad.

Keyboard Report

HID Keyboard Report Format

	BIT7	***BIT6***	***BIT5***	***BIT4***	***BIT3***	***BIT2***	***BIT1***	***BIT0***
Byte0	*Right GUI*	*Right Alt*	*Right Sft*	Right Ctrl	Left GUI	Left Alt	Left Shift	Left Ctrl
Byte1	*Reserved*							
Byte2	Key_array[0]							
Byte3	Key_array[1]							
Byte4	Key_array[2]							
Byte5	Key_array[3]							
Byte6	Key_array[4]							
Byte7	Key_array[5]							

For example if user has configured for Default Table. If the user now presses the top key, which is “<<” and USB code of 72. Then keyboard report sent to host would be:

Byte 0 – 0

Byte 1 – 0

Byte 2 – 72

Byte 3 – 0

Byte 4 – 0

Byte 5 – 0

Byte 6 – 0

Byte 7 – 0

Now if the user customizes the top key to be “R SHIFT” (modifier) and USB code for “a” (04). If the user presses the top key, then the keyboard report sent to host would be:

Byte 0 – 20 This is Right Shift modifier.

Byte 1 – 0

Byte 2 – 04

Byte 3 – 0

Byte 4 – 0

Byte 5 – 0

Byte 6 – 0

Byte 7 – 0

The API makes the following functions available to developers

Message Types

This is referenced in below functions:

```
REQUEST_TYPE{          // message types
DEVICE_STATUS = 1,     ///Device status message
LED_BRIGHTNESS,        //< set led brightness
RESERVED_1,           ///< not required for backward compatible
BUZZER_ON_OFF,         // Enable buzzer on/off
LOAD_NEW_TABLE,        //load new key code table
BUZZER_PERIOD,         // set buzzer on on for x ms
KEYPAD_TYPE,           // set keypad type
RESERVED_2,            //not required for backward compatible
WRITE_DEFAULT,         // Write defaults values from ram to flash
RESET_TO_FACTORY_DEFAULT, // reset the setting to factory default
RESERVED_3, //command to set number of keys installed 0-5keys, 1-     6keys, 2-8keys
RESERVED_4,            //start downloader
STATUS_BUZZER,         //Allows host app to sound the buzzer (0 - 9)
SET_SERIAL_NO,         // command to set serial number
RETRIEVE_JACK_STATE    //retrieves state of JACK i.e. JACK IN or JACK out }
```

InitialiseStormUSBDevice

This function is used to initialise the keypad. This is identified by the Product PID and Manufacturer VID. These are assigned to Keymat:

- Vendor ID – 0x2047
- Product ID – 0x09BF

On successful finding the keypad the manufacturer_local will be filled with “Storm Interface” and product_local will be filled with “EZ Key”. If not successful both of the strings will be filled with “none”

Parameters :

storm_vid	-	Vendor ID
product_pid	-	Product ID
manufacturer	-	vendors name will be stored
product	-	product name will be stored

Return Value:

True for success
False for failure.

```
///\brief InitializeStormUSBDevice is called at the beginning of the
application to

///Setup the PRODUCT ID (PID) and product vid

///\return false on failure, true on success.

///On failure, call GetErrorCode() to retrieve the error

///

bool  InitializeStormUSBDevice( int storm_vid, int product_pid);
```

GetDeviceStatus

This function retrieves status information about the NavPad™. For example, `led_brightness`, `buzzer_on_off`, `buzzer_period` etc. All information is stored in DEVICE_INFO structure.

Parameters :

```
typedef struct
{
      unsigned char           led_brightness;
      unsigned char           buzzer_on_off;
      unsigned char           buzzer_period;
      unsigned char           keypad_table;
      unsigned char           keyCode[20];
      std::string             version;
      std::string             serialNumber;
 } DEVICE_INFO;
```

_deviceInfo	-	DEVICE_INFO structure, that will be filled by the function
timeToWait	-	maximum time to wait for command to complete

Return Value:

True for success
False for failure.

```
///\brief GetDeviceStatus Retrieves the USB Display's status information including:
jack status, HV switch status, Firmware Name.
///The data are returned in  a DEVICE_INFO structure
///\param _deviceInfo is a pointer to a DEVICE_INFO structure that receives
information retrieved from the NavPad.
///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
///\return 0 on success, negative error code on failure
///

Int GetDeviceStatus( DEVICE_INFO *_deviceInfo, int _timeToWait );
```

SetLedLevel

This function sets the brightness of the LED. The led level can be set with values 0 to 9.

Parameters :

Int ledLevel
timeToWait - maximum time to wait for command to complete

Return Value:

0 for success

```
      ///\brief SetLedLevel This function sets led brightness level from 0 to 9,
where 0 is off
      ///                              and 9 is on.
      ///\param ledLevel    used to set led level
      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL          = User app passed in NULL
pointer for DEVICE_INFO structure
      ///         NO_USB_DISPLAY_CONNECTED                        = No keypad is
connected so cannot retrieve info
      ///         REQUEST_TIMEOUT                           = Could not retrieve the
info in the time alloted.
      ///
      DLLDEF int                                  SetLedLevel( int ledLevel, int
_timeToWait );
```

SetBuzzer

This function enables/disables the buzzer.

Parameters :

Int `enableBuzzer`
timeToWait - maximum time to wait for command to complete

Return Value:

0 for success

```
      ///\brief SetBuzzer This function enables/disables buzzer
      ///
      ///\param enableBuzzer        0 - disable 1 - enable
      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL          = User app passed in NULL
pointer for DEVICE_INFO structure
      ///         NO_USB_DISPLAY_CONNECTED                        = No keypad is
connected so cannot retrieve info
      ///         REQUEST_TIMEOUT                           = Could not retrieve the
info in the time alloted.
      ///
      DLLDEF int                                SetBuzzer( int enableBuzzer, int
_timeToWait );
```

LoadCodeTable

This function loads the keycode table in customise table.

Parameters :

Int *keyCodePtr	pointer to code table must hold 20 values including modifier. For each key the values must be [modifier, USB key code].
timeToWait	maximum time to wait for command to complete

Return Value:

0 for success

```
      ///\brief LoadCodeTable This function loads a new code table to customise
table in NavPad™
      ///
      ///\param KeyCodePtr - Point to new code table
      /// param keyCodeLen - length of keycode - Must be 20.
      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL          = User app passed in NULL
pointer for DEVICE_INFO structure
      ///         NO_USB_DISPLAY_CONNECTED                        = No keypad is
connected so cannot retrieve info
      ///         REQUEST_TIMEOUT                           = Could not retrieve the
info in the time alloted.
      ///
      DLLDEF int                                  LoadCodeTable( char *keyCodePtr, int
keyCodeLen, int _timeToWait );
```

SetBuzzerPeriod

This function sets the buzzer period from 0 (off) to 9 in increment of 0.25ms

Parameters :

Int buzzerPeriod	Buzzer period from 0 to 9
timeToWait	maximum time to wait for command to complete

Return Value:

0 for success

```
      ///\brief SetBuzzerPeriod This function sets buzzer period. How long buzzer
will sound for. Values are 0 to 9
      ///   0 - off 1 - 0.25ms 2 - .5ms ..
      ///\param buzzerPeriod        0 - 9
      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL          = User app passed in NULL
pointer for DEVICE_INFO structure
      ///         NO_USB_DISPLAY_CONNECTED                        = No keypad is
connected so cannot retrieve info
      ///         REQUEST_TIMEOUT                           = Could not retrieve the
info in the time alloted.
      ///
      DLLDEF int                                  SetBuzzerPeriod( int buzzerPeriod,
int _timeToWait );
```

SetKeypadTable

This function sets the current keypad code table that will be used. 0 – default, 1 – alternate, 2 - customise

Parameters :

Int KeypadTable	0 – default, 1 – alternate, 2 – customise
timeToWait	maximum time to wait for command to complete

Return Value:

0 for success

```
            ///\brief SetKeypadTable This function sets which table is currently
used.
      ///
      ///\param KeyCodeTable - 0 - default, 1 - alternate 2- customise

      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL          = User app passed in NULL
pointer for DEVICE_INFO structure
      ///         NO_USB_DISPLAY_CONNECTED                        = No keypad is
connected so cannot retrieve info
      ///         REQUEST_TIMEOUT                           = Could not retrieve the
info in the time alloted.
      ///
      DLLDEF int                                SetKeypadTable(int keyCodeTable, int
_timeToWait );
```

WriteDefaultToFlash

This function commands the NavPad™ to commit current values to flash.

Parameters :

timeToWait maximum time to wait for command to complete

Return Value:

0 for success

```
            ///\brief WriteDefaultToFlash This function writes changed values to
Flash
      ///
      ///\param None

      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL          = User app passed in NULL
pointer for DEVICE_INFO structure
      ///         NO_ NavPad™_CONNECTED         = No keypad is connected so cannot
retrieve info
      ///         REQUEST_TIMEOUT                     = Could not retrieve the info
in the time alloted.
      ///
      DLLDEF int                                WriteDefaultToFlash(int _timeToWait
);
```

ResetToFactoryDefault

This function commands the NavPad™ to reset the NavPad™ to factory default.

Parameters :

timeToWait	maximum time to wait for command to complete

Return Value:

0 for success

```
            ///\brief ResetToFactoryDefault This function reset AudioNav to factory
default
      ///
      ///\param None

      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL    = User app passed in NULL pointer
for DEVICE_INFO structure
      ///         NO_NavPad™_CONNECTED          = No keypad is connected so cannot
retrieve info
      ///         REQUEST_TIMEOUT               = Could not retrieve the info in the
time allotted.
      ///
      DLLDEF int                                ResetToFactoryDefault(int
_timeToWait );
```

GetJackStatus

This function command retrieves the status of the Jack

Parameters :

İnt *jackStatus	0 – Jack In 1 - Jack Out
timeToWait	maximum time to wait for command to complete

Return Value:

0 for success

```
      ///\brief GetJackStatus This function retrieves status of the JACK
      ///
      ///\param jackStatus

      ///\param _timeToWait is the time in milliseconds to wait for the data to be
retrieved.
      ///\return 0 on success, negative error code on failure
      ///     Possible error codes are:
      ///         DEVICE_INFO_STRUCTURE_NULL    = User app passed in NULL pointer
for DEVICE_INFO structure
      ///         NO_ NavPad™_CONNECTED         = No keypad is connected so cannot
retrieve info
      ///         REQUEST_TIMEOUT               = Could not retrieve the info in the
time allotted.
      ///
      DLLDEF int                                GetJackStatus(int *jackStatus, int
_timeToWait );
```

Remote Up
Change History

Instructions for	Date	Version	Details
API	15 Aug 24	1.0	First Release (split out from Tech Manual)

Host API Library	Date	Version	Details
	01 Sep 15	1.0	First Release
	08 Sep 17	7.0	Rolled in intermediate changes + Win 10 fix
	03 Apr 19	8.0	Added API command for Jack LED.