API Overview	2
Device Communications / Message Format	3
Message Definitions / Error Codes	5
Message Details	6
API Functions	13
Change History	24

API for controlling the AudioNav device from the Host Computer

This section provides details on how the AudioNav can be controlled from a host that has USB capabilities.

Host API Library - Overview

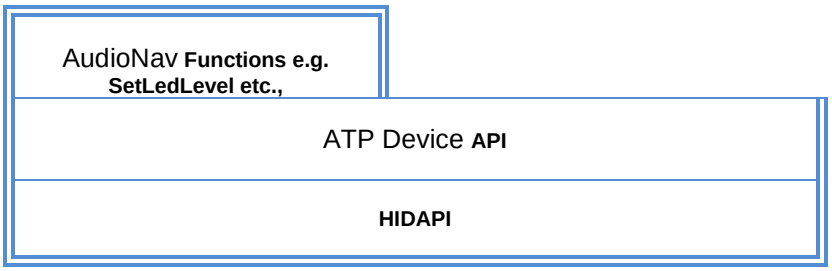
The Host API Library is a middleware application between the host application and the AudioNav. This is available as a download together with the HIDAPI library.

- The API library allows for the host application to invoke the AudioNav 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 AudioNav 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.

List of Messages

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

ID	Name	Description
01	Device Status Request	Output the firmware version & selected parameters
02	LED Brightness	Adjust led brightness.
03	Reserved	Reserved for future use
04	Reserved	Reserved for future use
05	Load New code table	Load new code table
06	Reserved	Reserved for future use
07	Keypad Type	Select layout table
08	Reserved	Reserved
09	Write to default	AudioNav writes configuration data from ram to flash
10	Reset to factory default	Reset device back to factory default
11	Reserved	Reserved for future use
12	Load Firmware	Sets the AudioNav to detect the device loader for firmware loading
13	Reserved	Reserved for future use
14	Set Serial Number	Write 12 digit serial number

Structure of Messages from AudioNav to Host

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

AudioNav Device Communications

AudioNav 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

Message ID Definitions

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 AUDIONAV keypad – Output the firmware version and all currently selected parameters
02	LED Brightness	Host To AUDIONAV keypad – adjust led brightness. (default: 0)
03	Reserved	RESERVED
04	Reserved	RESERVED
05	Load New code table	Host To AUDIONAV keypad – Load new code table
06	Reserved	RESERVED
07	Keypad Table	Host To AUDIONAV keypad – Select layout table 0 – Default Table 1 – Alternate Table 2 – Customised
08	Reserved	Reserved
09	Write to default	Host To AUDIONAV – AudioNav writes configuration data from ram to flash.
10	Reset to factory default	Host To AUDIONAV – Reset device back to factory default
11	Reserved	RESERVED
12	Load Firmware	Host To AUDIONAV– Sets the AudioNav to detect the device loader for firmware loading
13	Reserved	RESERVED
14	Set Serial Number	Host to AUDIONAV– to store a serial number (12 digits)

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

Device Status (01)

Host sends this message to AudioNav to request the status of the AudioNav keypad

Host Device		AUDIONAV-
[01]	⇨	
	⇦	[01][ec][lb][kt][Js][Hs][20*kc][fw][sn]

AudioNav **Status Response**

Keypad 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)
Kt	Keypad Table	SN	1	0 – Default Table 1 – Alternate Table 2 – Customised Table
Js	Jack status	SN	1	0 – Jack IN, 1 – Jack Out
Hs	Horizontal/Verticle	SN	1	0 – Vertical 1 - Horizontal
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 “
sn	Serial Number	ANS	12	Returns serial number YYQQXXXXXXXX Where YY – year, QQ – Quarter XXXXXXXX – Sequential number

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		AudioNav
[02][lb]	⇨	
	⇦	[02][ec]

Note: LED brightness of 0 value indicates LEDs are off

LED brightness of 9 value indicates full brightness

Reserved (03)

Reserved (04)

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		AudioNav
[05][lt][20 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

Reserved (06)

Keypad Table Command (07)

Host sends this message to set keypad type

	Data Field	Type	Length	Description	
1	Keypad Type	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		AudioNav
[07][bp]	⇒	
	⇐	[07][ec]

Reserved (08)

Write Config Data To Flash command (09)

Host sends this command to request the AUDIONAV to write the configuration data from RAM to FLASH. This command has no data associated with it.

Host Device		AUDIONAV-
[09]	⇒	
	⇐	[09][ec]

RAM to FLASH command **Response**

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Reset To Factory Default command (10)

Host sends this command to request the AUDIONAV to reset parameters back to factory default. This command has no data associated with it.

Reset To Factory Default **Response**

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		AUDIONAV-
[10]	⇒	
	⇐	[10][ec]

Reserved (11)

Enable BSL Command (12)

Host sends this command to request the AUDIONAV to start downloader

Enable BSL command **Response**

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		AUDIONAV-
[12]	⇒	
	⇐	[12][ec]

Reserved (13)

Set Serial Number command (14)

Host sends this command to set the serial number of the device in format YYQQXXXXXXXX

	Data Field	Type	Length	Description
1	Serial Number	ANS	12	YYQQXXXXXXXX

Set Serial Number command **Response**

	Data Field	Type	Length	Description
ec	Error Code	H	2	

Host Device		AUDIONAV-
[14][SN]	⇒	
	⇐	[14][ec]

(01) Key Press Code

With the USB stack configured for a standard keyboard interface, the AudioNav 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

Page

All Message Types 14

GetDeviceStatus 16

InitialiseStormUSBDevice 15

LoadCodeTable 18

ResetToFactoryDefault 21

SetKeypadTable 19

WriteDefaultToFlash 20

Message Types

This is referenced in below functions:

```
    enum REQUEST_TYPE{      // message types

    DEVICE_STATUS = 1,              ///Device status message
    LED_BRIGHTNESS,                 //< set led brightness
    RESERVED_1,                           ///MID_RESERVED_6
    RESERVED_2,                     // MID_RESERVED_6
    LOAD_NEW_TABLE,                       //load new key code table
    RESERVED_3,                // MID_RESERVED_6
    KEYPAD_TYPE,                    // set keypad type 0 - default table, 1 -
alternate 2- customise
    RESERVED_4,                           //MID_RESERVED_6
    WRITE_DEFAULT,                      // Write defaults values from ram to flash
    RESET_TO_FACTORY_DEFAULT,    // reset the setting to factory default
    RESERVED_5,                     //MID_RESERVED_6
    ENABLE_BSL,                     //start downloader
    RESERVED_6                 //MID_RESERVED_6

      }
```

InitialiseStormUSBDevice

This function is used to initialise the Audio Nav. The Audio Nav is identified by the Product PID and Manufacturer VID. This are assigned to Keymat:

- Vendor ID – 0x2047
- Product ID – 0x09D0

On successful finding the Audio Nav the manufacturer_local will be filled with “Storm Interface” and product_local will be filled with “AUDIO NAV”. 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 Audio Nav. For example, Jack status, HV switch status, led level status etc. All information is stored in DEVICE_INFO structure.

Parameters :

```
typedef struct
{
      unsigned char           led_brightness;
      unsigned char           keypad_table;
      unsigned char           jack_status;
      unsigned char           HV_status;
      unsigned char           keyCode[20]; //currently keytable in use
      std::string             version;
      std::string             serialNumber;
} DEVICE_INFO;
```

_deviceInfo	-	DEVICE_INFO sturcture, 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 Audio Nav.
///\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 );
```

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 AudioNav
	///
	///\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 );
```

SetKeypadTable

This function sets the current keypad 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 commnds the AudioNav 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_USB_DISPLAY_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 commnds the AudioNav to reset the Audio Nav 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_USB_DISPLAY_CONNECTED                        = No keypad is
connected so cannot retrieve info
      ///         REQUEST_TIMEOUT                           = Could not retrieve the
info in the time alloted.
      ///
      DLLDEF int                                  ResetToFactoryDefault(int
_timeToWait );
```

Change History

Instructions for	Date	Version	Details
API	15 Aug 2024	1.0	First release (split out from combined document)

Host API Library	Date	Version	Details
	01 Sep 15	1.0	First Release
	08 Sep 17	4.0	Added Win 10 Compatibility
	05 Feb 21	7.0	Update to Visual Studio latest
	08 Mar 24	10.0	Added components for downloader utility