ENGAUGE

ENGAUGE LIBRARY API SPECIFICATION

1. Introduction

The Engauge HID library provides an API for communication with an Engauge Device. This library provides methods for extracting device information and sending and receiving Engauge HID reports. C libraries are provided for Windows 2000 and later.

2. Library Usage

The Engauge Library contains the following files:

- Engauge.dll – Dynamic link library that exports a C Engauge library
- Engauge.lib—Provides build-time linking for C/C++ projects
- Engauge.h—Defines Engauge library function prototypes and constants

2.1. C/C++ Projects

The Engauge Library is designed to be universally compatible with most programming languages and is exported in C.

Perform the following steps to use the library in a C/C++ project:

1. Add “Engauge.lib” as a linker input dependency or add the following code:
   #pragma comment (lib, “Engauge.lib”)
2. Include “Engauge.h” as needed.
3. Provide “Engauge.dll” with the executable (.exe file)

3. Engauge Library Functions

Table 1. API Function Table

Definition	Description	Page #
Engauge_Enumerate()	Returns the number of Engauge devices connected	2

3.1. Engauge_Enumerate

Description: This function returns the number of Engauge devices that are connected to the host.

Prototype: DWORD Engauge_Enumerate(void)

Parameters: None

Return Value: A return value of 0 indicates that no devices are available. Otherwose this function returns the number of connected Engauge devices. When referring to a device by *deviceIndex* , the index may range from 0 to (Engauge_Enumerate() – 1).

Remarks: This function returns the number of connected Engauge devices. This does not imply that a device is available for use. Users must call Engauge_Enumerate() before calling any function that takes a device index as a parameter in order to build an up-to-date device list. If a device is installed or removed after calling Engauge_Enumerate(), then the device list will be out of date, and the results may be unpredictable. Currently, Engauge_Open() is the only function that takes a device index parameter.

4. Engauge Library Functions for Opened Devices

Table 2. API Functions Table

Definition	Description	Page #
Engauge_Open()	Opens an Engauge using a device index	3
Engauge_Close() ()	Closes the currently open Engauge device	3
Read_Engauge_IDs()	Receives a list of Engauge IDs	4
Set_Pointer_LED_State()	Sets the selected Engauge pointer LED state	5
Set_Backlight_Color()	Sets the selected Engauge backlight color	6
Set_Pointer_Posistion()	Sets the selectd Engauge pointer position	7
Set_All_Pointer_LED_State()	Sets all Engauge pointer LEDs to the same state	8
Set_All_Backlight_Color()	Sets all Engauge backlights to the same color	9
Set_All_Pointer_Position()	Sets all Engauge pointers to the same position	10
Note: These functions requite an additional “hid_device device” parameter at the beginning of the argument list. This parameter is an HID class object point as returned by Engauge_Open().		

4.1 Engauge_Open

Description: Opens an Engauge device using a device index and returns a device object pointer which will be used for subsequent access.

Prototype: `BYTE Engauge_Open (HID_DEVICE **device, unsigned short deviceIndex )`

Parameters: 1. `device`—A pointer to an HID class object pointer used for subsequent device access.

2. `deviceIndex` – Zero-based device index ranging from 0 to (`Engauge_Enumerate()` -1)

Return Value: 0 - Success
1 – No device found
2 – Device already open

Example:
```
hid_device *handle = NULL;
BYTE bretval;

bretval = Engauge_Open(&handle, 0);
```

Remarks: None

4.2 Engauge_Close

Description: This function closes the currently-open Engauge device.

Prototype: `BYTE Engauge_Close (HID_DEVICE *device)`

Parameters: 1. `device`—A pointer to an HID class object returned by `Engauge_Open()`.

Return Value None

Example:
```
BYTE bretval;

bretval = Engauge_Close(handle);
```

Remarks: This function deallocates the HID class object allocated in Engauge_Open().

4.3 Read_Engauge_IDs

Description: This function sends an HID output report from the host to a currently-open the Engauge device over the interrupt endpoint. This report issues a request to the Engauge device, and waits for it to send back, via an input report, a list of Engauge device IDs for all Engauge devices attached to the particular USB port.

Prototype: `int Read_Engauge_IDs(hid_device *device, Engauge_ID **Ids)`

Parameters: 1. `device`—A pointer to an HID class object returned by Engauge_Open().

2. `Ids` — A pointer to a linked list structure pointer used to hold all ID's of devices connected to the Engauge *device*, including itself.

Return Value This function returns 0 on success and -1 on error.

Example:

```
Engauge_ID *ids;
int iretval;

iretval = Read_Engauge_IDs(handle, &ids);
```

Remarks: Use this function to build a linked list of Engauge IDs for all Engauge devices attached to a USB port. The built linked list is of the form:

```
Struct Engauge_ID {
    unsigned int engauge_id;
    struct Engauge_ID *next;
};
```

4.4 Set_Pointer_LED_State

Description: This function sends an HID output report from the host to a currently-open Engauge device over the interrupt endpoint. Use this function to turn on or off the pointer LEDs of a specific Engauge device.

Prototype: `int Set_Pointer_State(hid_device *device, unsigned int uiDevice_ID, unsigned char bState)`

Parameters: 1. device—A pointer to an HID class object returned by Engauge_Open().

2. Ids — The ID of the specific targeted Engauge.

3. bState—The state of the Pointer LEDs (1= on, 0 = off)

Return Value This function returns 0 on success and -1 on error.

Example:

```
int iretval;

// The following code will turn on the pointer LEDs to the first
// Engauge device in an "chain" of Engauge devices.

iretval = Set_Pointer_LED_State(handle, ids->engauge_id, 1);
```

Remarks: None.

4.5 Set_Backlight_Color

Description: This function sends an HID output report from the host to a currently-open Engauge device over the interrupt endpoint. Use this function to change the backlight color of a specific Engauge device.

Prototype: `int Set_Backlight_Color(hid_device *device, unsigned int uiDevice_ID, unsigned int uiColor)`

Parameters: 1. `device`—A pointer to an HID class object returned by `Engauge_Open()`.

2. `Ids` — The ID of the specific targeted Engauge.

3. `uicolor`—The RGB value of the Engauge backlight.

Return Value This function returns 0 on success and -1 on error.

Example:

```
int iretval;

// The following code will turn the backlight of first Engauge device
// in a "chain" of Engauge devices to the color white

iretval = Set_Pointer_LED_State(handle, ids->engauge_id, 0xFFFFFF);
```

Remarks: None.

4.6 Set_Pointer_Position

Description: This function sends an HID output report from the host to a currently-open Engauge device over the interrupt endpoint. Use this function to set the pointer location of a specific Engauge device.

Prototype: `int Set_Pointer_Position(hid_device *device, unsigned int uiDevice_ID, unsigned int uiPosition)`

Parameters: 1. `device`—A pointer to an HID class object returned by `Engauge_Open()`.

2. `Ids` — The ID of the specific targeted Engauge.

3. `uiPosition`—The location of the Engauge pointer in micro-steps (0-3204 micro-steps are available).

Return Value This function returns 0 on success and -1 on error.

Example:

```
int iretval;

// The following code positions the pointer of first Engauge device in
// a "chain" of Engauge devices

iretval = Set_Pointer_LED_State(handle, ids->engauge_id, 1602);
```

Remarks: The Engauge pointer has a full sweep range of 267 angular degrees, and there are 12 micro positions in each 1 degree. Thus the pointer of the Engauge can be placed in any of 3204 positions. By way of example if your Engauge dial indicates 0 -100 degrees Celsius and you wanted to position the Engauge pointer at 50 degrees Celsius you would use the following code which would move the pointer to a micro-step position of 1602, ½ of the full position of 3204 micro-steps;

```
iretval = Set_Pointer_LED_State(handle, ids->engauge_id, 1602);
```

4.7 Set_All_Pointer_LED_State

Description: This function sends an HID output report from the host to a currently-open Engauge device over the interrupt endpoint. Use this function to turn on all Engauge device pointers attached to a USB port.

Prototype: `int Set_All_Pointer_State(hid_device *device, unsigned char bState)`

Parameters: 1. device—A pointer to an HID class object returned by Engauge_Open().

2. bState—The state of the Pointer LEDs (1= on, 0 = off)

Return Value This function returns 0 on success and -1 on error.

Example:

```
int iretval;

// The following code will turn on the pointer LEDs to the first
// Engauge device in an "chain" of Engauge devices.

iretval = Set_All_Pointer_LED_State(handle, 1);
```

Remarks: None.

4.8 Set_All_Backlight_Color

Description: This function sends an HID output report from the host to a currently-open Engauge device over the interrupt endpoint. Use this function to change the backlight color all Engauge devices attached to a USB port.

Prototype: `int Set_All_Backlight_Color(hid_device *device, unsigned int uiColor)`

Parameters: 1. `device`—A pointer to an HID class object returned by `Engauge_Open()`.

2. `uicolor`—The RGB value of the Engauge backlight.

Return Value This function returns 0 on success and -1 on error.

Example:

```
int iretval;

// The following code will turn the backlight of all Engauge devices
// in a "chain" color white

iretval = Set_All_Pointer_LED_State (handle, 0xFFFFFF);
```

Remarks: None.

4.9 Set_ALL_Pointer_Position

Description: This function sends an HID output report from the host to a currently-open Engauge device over the interrupt endpoint. Use this function to set the pointer locations of all Engauge devices attached to a USB port.

Prototype: `int Set_All_Pointer_Position(hid_device *device, unsigned int uiPosition)`

Parameters: 1. `device`—A pointer to an HID class object returned by `Engauge_Open()`.

2. `uiPosition`—The location of the Engauge pointer in micro-steps (0-3204 micro-steps are available).

Return Value This function returns 0 on success and -1 on error.

Example:

```
int iretval;

// The following code positions all pointers of and Engauge "chain" to
// full sweep.

iretval = Set_All_Pointer_LED_State (handle, 1602);
```

Remarks: None