Logitech PWS&G: Samarkand Docs Logout

x19c0 - Force Sensing Button

Version 1

Force Sensing Button

This feature allows a configuration of the force sensing button.

[0] **getCapabilities**() → numButtons
[1] **getButtonCapabilities**(button_id) → capabilities, defaultForce, maxForce, minForce, numOfThresholds
[2] **getButtonConfig**(button_id) → l1_threshold, l2_threshold
[3] **setButtonConfig**(button_id, l1_threshold, l2_threshold) → button_id, l1_threshold, l2_threshold
[4] **resetButtonConfig**(button_id) → l1_threshold, l2_threshold

Overview

This feature covers all type of sensing buttons regardless of the technology used to detect the force applied on the button (Inductive, Capacitive, etc.). It gives the possibility to change the force applied to trigger the button action.

Functions and Events

[0] getCapabilities() → numButtons

Get device general capabilities.

Parameters

none

Returns

numButtons

[1 byte] Number of buttons supported by the device.

Table 1. getCapabilities() response packet format

byte \ bit	7	6	5	4	3	2	1	0
0	numButtons							
1..15	*reserved*							

[1] getButtonCapabilities(button_id) → capabilities, defaultForce, maxForce, minForce, numOfThresholds

Returns the supported sensing button capabilities.

Parameters

button_id

[1 byte] Button ID (0..numButtons-1). The Force buttons are indexed from **right** to **left** (and then **furthest** to **closest**) as seen by the user.

Table 2. getButtonCapabilities() request packet format

byte \ bit	7	6	5	4	3	2	1	0
0	button_id							
1..15	*reserved*							

Returns

capabilities

[2 bytes] Supported capabilities

```
bit 0: force_change, customizable force (0: Not supported, 1: supported). The
force value can be changed by the user in a predefined range.
bit 1..15: reserved
```

defaultForce

[2 bytes] Default force value to trigger the button, for multiple thresholds this is the default first threshold value.

maxForce

[2 bytes] Maximum force value to trigger the button.

minForce

[2 bytes] Minimum force value to trigger the button.

numOfThresholds

[1 byte] Number of thresholds supported by the button; Minimum should be 1. Maximum should be 2.

NOTE

The force unit is a count ADC value.

Table 3. getButtonCapabilities() response packet format

byte \ bit	7	6	5	4	3	2	1	0
0	capabilities (MSB)							
	reserved							
1	capabilities (LSB)							
	reserved							force_change
2	defaultForce (MSB)							
3	defaultForce (LSB)							
4	maxForce (MSB)							
5	maxForce (LSB)							
6	minForce (MSB)							
7	minForce (LSB)							
8	numOfThresholds							
9..15	*reserved*							

Errors

- HIDPP_ERR_INVALID_ARGUMENT: if "button_id" is out of range.

[2] getButtonConfig(button_id) → I1_threshold, I2_threshold

The function returns the current button configuration.

Parameters

button_id

[1 byte] Button ID (0..numButtons-1).

Table 4. getButtonConfig() request packet format

byte \ bit	7	6	5	4	3	2	1	0
0	button_id							
1..15	*reserved*							

Returns

I1_threshold

[2 bytes] The current L1 threshold value (in ADC count) to trigger the button and its relevant functionality; the L1 threshold is the first threshold value for the button, and it will always have a value (even if the device supports single threshold).

I2_threshold

[2 bytes] The current L2 threshold value (in ADC count) to trigger the button and its relevant functionality; when device support with only single threshold, this value will return with zero.

Table 5. getButtonConfig() response packet format

byte \ bit	7	6	5	4	3	2	1	0
0	I1_threshold (MSB)							
1	I1_threshold (LSB)							
2	I2_threshold (MSB)							
3	I2_threshold (LSB)							
4..15	*reserved*							

Errors

- HIDPP_ERR_INVALID_ARGUMENT: if "button_id" is out of range.

[3] setButtonConfig(button_id, I1_threshold, I2_threshold) → button_id, I1_threshold, I2_threshold

Set the configuration in a persistent manner (written in NVM).

Parameters

button_id

[1 byte] Button ID (0..numButtons-1).

I1_threshold

[2 bytes] The new L1 Threshold value mentioned in the definition of getButtonConfig; this is the first threshold value for the button, and it will always have a value (even if the device supports single threshold). The value must be in the range [minForce..maxForce] and cannot be zero.

l2_threshold

[2 bytes] The new L2 Threshold value mentioned in the definition of getButtonConfig; when numOfThresholds returned by getButtonCapabilities is 1, this value will be ignored; when numOfThresholds is 2, this value must be in the range [minForce..maxForce] and l2_threshold must be greater than l1_threshold or HIDPP_ERR_INVALID_ARGUMENT will be returned.

Table 6. setButtonConfig() request packet format

byte \ bit	7	6	5	4	3	2	1	0
0	button_id							
1	l1_threshold (MSB)							
2	l1_threshold (LSB)							
3	l2_threshold (MSB)							
4	l2_threshold (LSB)							
5..15	*reserved*							

Returns

button_id

[1 byte] Echo of the selected Button ID.

l1_threshold

[2 bytes] Echo of the current applied value.

l2_threshold

[2 bytes] Echo of the current applied value; return zero if the device supports only single threshold.

NOTE

This value will be set by the FW. It can be slightly different from the requested value if for example the device does not support all force granularities.

Table 7. setButtonConfig() response packet format

byte \ bit	7	6	5	4	3	2	1	0
0	button_id							

byte \ bit	7	6	5	4	3	2	1	0
1	I1_threshold (MSB)							
2	I1_threshold (LSB)							
3	I2_threshold (MSB)							
4	I2_threshold (LSB)							
5..15	*reserved*							

Errors

- HIDPP_ERR_INVALID_ARGUMENT:
  - if "I1_threshold" is out of range [minForce..maxForce] or set to zero.
  - When device support multiple threshold, if "I2_threshold" is out of range [minForce..maxForce] and I2_threshold is less or equal to I1_threshold.
  - if "button_id" is out of range.
  - in case of a mismatch between capabilities.
- HIDPP_ERR_NOT_ALLOWED: if this function is called and the "force_change" is not supported for the selected button.

[4] resetButtonConfig(button_id) → I1_threshold, I2_threshold

This function resets the button configuration to the default values stored in the device NVM. It is useful when the user wants to restore the button to its factory settings.

Parameters

button_id

[1 byte] Button ID (0..numButtons-1).

Table 8. resetButtonConfig() request packet format

byte \ bit	7	6	5	4	3	2	1	0
0	button_id							
1..15	*reserved*							

Returns

I1_threshold

[2 bytes] The current default L1 Threshold value mentioned in the definition of getButtonConfig; this is the first threshold value for the button, and it will always have a value (even if the device supports single threshold).

l2_threshold

[2 bytes] The current default L2 Threshold value mentioned in the definition of getButtonConfig; when numOfThresholds returned by getButtonCapabilities is 1, this value will return with zero.

Table 9. resetButtonConfig() response packet format

byte \ bit	7	6	5	4	3	2	1	0
0	l1_threshold (MSB)							
1	l1_threshold (LSB)							
2	l2_threshold (MSB)							
3	l2_threshold (LSB)							
4..15	*reserved*							

Errors

- HIDPP_ERR_INVALID_ARGUMENT: if "button_id" is out of range.

ChangeLog

- Version 1: Added support for multiple thresholds, update getButtonCapabilities, getButtonConfig, setButtonConfig to support multiple thresholds; Maximum number of threshold layers is 2 for this version; add resetButtonConfig function.
- Version 0: Initial version