

Commands.

1. Application → InputStick:

<i>CMD:</i>	Required:	FW >=
Run firmware (0x04)	Y	v0.94
Get firmware info (0x10)	Y	v0.94
Reset report buffers (0x11)	N	v0.94
Authenticate (0x12)	Y	v0.94
Settings (0x14)	N	v0.94
Restore defaults (0x15)	N	v0.94
Read settings (0x17)	N	v0.97
Set PIN (0x18)	N	v0.97
USB Resume (0x19)	N	v0.97
USB Power (0x1A)	N	v0.97
Unlock (0x1B)	N	v0.97
Queue keyboard reports (0x21)	Y	v0.94
Queue consumer reports (0x22)	Y	v0.94
Queue mouse reports (0x23)	Y	v0.94
Queue gamepad reports (0x24)	N	v0.97
Queue mixed reports (0x25)	N	v0.98
Write to endpoint (0x2B)	N	v0.97
Queue SHORT keyboard reports (0x2C)	N	v0.97
Queue Press and Release events (0x2D)	N	v0.97

Commands that are not marked as “Required” are not necessary for normal operation. Setting up password protection or restoring factory defaults can be performed using Android smartphone or tablet (InputStickUtility application).

“Authentication required”: applies only when password protection is enabled. If not, all commands can be executed without authentication.

Run firmware (0x04)

Short description:	Run firmware	
<i>CMD</i> byte:	0x04	
Authentication required:	No	
Since FW version:	0.94	
<i>PARAM</i> :	None (0x00)	
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
<i>RESP_DATA</i> :	None	
Details:	When this command is received by bootloader, it will immediately jump to main firmware. When it is received by main firmware, nothing happens. Always use this command after establishing connection.	

Usage example: see (6) examples.pdf.

Get firmware info (0x10)

Short description:	Requests firmware information	
<i>CMD</i> byte:	0x10	
Authentication required:	No	
Since FW version:	0.94	
<i>PARAM</i> :	None (0x00)	
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
<i>RESP_DATA</i> :	Bytes:	Description:
	0	Firmware type ('B')
	1	Firmware version, major (0x00 = 0)
	2	Firmware version, minor (0x60 = 96)
	3	Hardware revision (0x01 = 1)
	4	Reserved
	5	Reserved
	6	USB pullup (0x01 – pullup enabled, 0x00 pullup disabled)
	7	USB state (0x05 – USB ready)
	8	Startup status (0x08 - OK)
	9..15	Reserved
	16	Encrypt outgoing data
	17	Authenticated
	18	Password protected
	19..23	Reserved
	23..27	System time (ms)
	28..31	Total RX bytes
	32..35	Total TX bytes
Details:	Use this command to identify hardware and firmware version. Always use this command before sending any command that requires authentication. If password protection is enabled, it will be necessary to authenticate first.	

Notes:

USB pullup:

Value:	State:	Description:
0x00	Disabled	Pullup disabled. USB host will assume the device is disconnected.
0x01	Enabled	Pullup enabled. USB host will be able to detect the device.

USB state:

Value:	State:	Description:
0x00	Unconnected	Initial state.
0x01	Attached	State after bus reset.
0x02	Powered	
0x03	Suspended	
0x04	Addressed	Host assigned address to the InputStick device.
0x05	Configured (READY)	Host is ready to receive data (reports).

Usage example: see (6) examples.pdf.

Reset report buffers (0x11)

Short description:	Resets state of report buffers	
<i>CMD</i> byte:	0x11	
Authentication required:	Yes	
Since FW version:	0.94	
<i>PARAM</i> :	None (0x00)	
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
<i>RESP_DATA</i> :	None	
Details:	All reports still present in buffers will be removed. This command is automatically executed on system startup, it is no longer necessary to send it during initialization.	

Authenticate (0x12)

Short description:	Perform authentication	
CMD byte:	0x12	
Authentication required:	No	
Since FW version:	0.94	
PARAM:	Enable outgoing encryption (currently ignored!)	
DATA:	Bytes:	Description:
	0..15	AES-128 IV (Initialize Vector)
	16..31	KeyVerificationData
	32..47	KeyChallengeData
RESP_CODE:	0x01	OK
	0x20	Invalid encryption key
	0x21	Password protection is not enabled (no encryption key present)
RESP_DATA:	Bytes:	Description:
	0..15	KeyResponseData (encrypted KeyChallengeData)
Details:	<p><i>KeyVerificationData</i> – allows InputStick to verify if remote application knows the encryption key.</p> <p><i>KeyChallengeData</i> – used by InputStick to generate <i>KeyResponseData</i>.</p> <p><i>KeyResponseData</i> – used by InputStick to prove that it knows the encryption key.</p> <p>How to generate <i>KeyVerificationData</i>:</p> <ol style="list-style-type: none"> 1. Reserve 4 bytes for CRC32 value. 2. Generate 12 random bytes. 3. Calculate CRC32 checksum using previously generated data. 4. Store CRC32 data into previously reserved space or leave 0x00000000 (not recommended). 5. Encrypt all 16 bytes. <p>InputStick will decrypt received <i>KeyVerificationData</i>, calculate CRC32 and compare with provided value. If encryption keys are identical on both sides, CRC32 values will be identical.</p> <p>How to generate <i>KeyChallengeData</i>:</p> <ol style="list-style-type: none"> 1. Generate 16 random bytes. <p>What to do with <i>KeyResponseData</i>?</p> <ol style="list-style-type: none"> 1. Decrypt <i>KeyResponseData</i>. 	

	<ol style="list-style-type: none">2. Compare result with previously sent <i>KeyChallengeData</i>.3. Both arrays must be identical.
--	---

Usage example: see (6) examples.pdf.

Settings (0x14)

Short description:	Set Settings	
<i>CMD</i> byte:	0x14	
Authentication required:	No	
Since FW version:	0.94	
<i>PARAM</i> :	0x01	Set password
<i>DATA</i> :	Depends on <i>PARAM</i> .	
<i>RESP_CODE</i> :	0x01	OK
	0x05	Invalid Settings code (<i>PARAM</i>)
<i>RESP_DATA</i> :	Bytes:	Description:
	0..15	<i>KeyResponseData</i> (encrypted <i>KeyChallengeData</i>)
Details:	When settings password (<i>PARAM</i> = 0x01), <i>DATA</i> is a AES-128 key (16B). When key is set to 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF, password protection is disabled.	

Restore defaults (0x15)

Short description:	Restore system defaults.	
<i>CMD</i> byte:	0x15	
Authentication required:	No	
Since FW version:	0.94	
<i>PARAM</i> :	0x00	Cancel restore procedure.
	0x01	Start restore procedure.
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
	0xFE	Invalid <i>PARAM</i>
<i>RESP_DATA</i> :	None	
Details:	<p>Initiates „Restore defaults” procedure. User must set requested state of keyboard LEDs (NumLock, CapsLock, ScrollLock) several times. When completed successfully, InputStick will be restored to factory defaults (password protection will be removed).</p> <p>See also: Restore defaults Update (0x16)</p>	

Read settings (0x17)

Short description:	Reads settings specified by <i>PARAM</i> .	
<i>CMD</i> byte:	0x17	
Authentication required:	No	
Since FW version:	0.97	
<i>PARAM</i> :	0x02	Reads USB descriptor settings
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
	0xFE	Invalid <i>PARAM</i>
<i>RESP_DATA</i> :	Settings data (specified by <i>PARAM</i>)	
Details:	Length of <i>RESP_DATA</i> : 18B (<i>PARAM</i> = 0x02, USB descriptor configuration).	

Set PIN (0x18)

Short description:	Sets Bluetooth pairing PIN	
<i>CMD</i> byte:	0x18	
Authentication required:	No	
Since FW version:	0.97	
<i>PARAM</i> :	0x04	4 digit PIN (BT2.1 devices only)
	0x06	6 digit PIN (BT4.0 devices only)
<i>DATA</i> :	PIN digits (ASCII)	
<i>RESP_CODE</i> :	0x01	OK
	0x26	Invalid character (not a ASCII digit)
	0x27	PIN change is already in progress
	0x28	Invalid number of characters
<i>RESP_DATA</i> :	None	
Details:	<p>1 second after sending response, InputStick will reset BT module, connection will be lost. PIN will be changed 2 seconds later. Success will be indicated by toggling keyboard LEDs (Windows only).</p> <p>After successful PIN change InputStick must be removed from USB port and paired again during next connection attempt.</p> <p>Setting PIN for BT4.0 devices is not recommended due to bug in Android OS (OS will ask user to provide PIN during each connection attempt).</p>	

USB Resume (0x19)

Short description:	Resumes USB host from sleep mode.	
<i>CMD</i> byte:	0x19	
Authentication required:	No	
Since FW version:	0.97	
<i>PARAM</i> :	None	
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
<i>RESP_DATA</i> :	None	
Details:	Attempts to resume USB host from sleep or suspend mode. Will not work if USB wake-up is not supported or not enabled.	

USB Power (0x1A)

Short description:	Sets USB pullup resistor on/off.	
<i>CMD</i> byte:	0x17	
Authentication required:	No	
Since FW version:	0.97	
<i>PARAM</i> :	0x00	Pullup resistor disabled (simulates disconnecting USB device).
	0x01	Pullup resistor enabled (device is detected by USB host).
	0x02	Disable pullup, enable after 3 seconds. Device will be enumerated again.
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
<i>RESP_DATA</i> :	None	
Details:	Enabled or disables USB pullup resistor. Can be used to simulate disconnect event.	

Unlock (0x1B)

Short description:	Requests to unlock InputStick.	
<i>CMD</i> byte:	0x1B	
Authentication required:	No	
Since FW version:	0.97	
<i>PARAM</i> :	None	
<i>DATA</i> :	None	
<i>RESP_CODE</i> :	0x01	OK
	0x25	Unlocking not possible (timed-out)
<i>RESP_DATA</i> :	None	
Details:	<p>No need to manually send this command. InputStick is automatically unlocked for first 30 seconds after powering up. If password protection is set it is always in unlocked state.</p> <p>This command allows to check if InputStick is in unlocked state (0x01 <i>RESP_CODE</i>)</p> <p>See Security chapter for more details.</p>	

Queue keyboard reports (0x21)

Short description:	Puts HID keyboard reports into buffer.	
CMD byte:	0x21	
Authentication required:	Yes	
Since FW version:	0.94	
PARAM:	Number of HID keyboard reports in packet (1 - 32)	
DATA:	PARAM * HID keyboard reports (Report1, Report2, ... ReportPARAM)	
RESP_CODE:	0x01	OK
	0x03	Buffer overflow (no reports were put into buffer!)
RESP_DATA:	None	
Details:	As long as you keep track of number of HID reports currently stored in buffer, there is no need to request response to this command. See HID chapter for keyboard report details.	

Example:

Type „A” character: press left Shift key, hold left Shift key while pressing „A” key, release all keys.

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	HID Keyboard Report 1								HID Keyboard Report 2					
0x21	0x03	0x02	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x02	0x00	0x04	0x00	0x00	0x00

...	HID Keyboard Report 3								
0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x00

CRC32: 4B

Payload: 26B

Add 2B padding (0x00...)

Total length = 32B

Header (no response, no encryption): 0x55, 0x02

Remarks:

In most cases you should always put empty keyboard report (0x00, .. 0x00 – all keys released) each time you queue keyboard reports. This allows to prevent from situation when some key remains in “pressed” state, when Bluetooth connection is lost before queuing another set of reports.

Queue consumer reports (0x22)

Short description:	Puts HID consumer reports into buffer.	
<i>CMD</i> byte:	0x22	
Authentication required:	Yes	
Since FW version:	0.94	
<i>PARAM</i> :	Number of HID consumer reports in packet (1 - 32)	
<i>DATA</i> :	<i>PARAM</i> * HID consumer reports (Report1, Report2, ... Report <i>PARAM</i>)	
<i>RESP_CODE</i> :	0x01	OK
	0x03	Buffer overflow (no reports were put into buffer!)
<i>RESP_DATA</i> :	None	
Details:	As long as you keep track of number of HID reports currently stored in buffer, there is no need to request response to this command. See HID chapter for consumer report details.	

Example:

Increase system volume:

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	HID Consumer Report 1			HID Consumer Report 2		
0x23	0x02	0x01	0x00	0xE9	0x01	0x00	0x00

CRC32: 4B

Payload: 8B

Add 4B padding (0x00...)

Total length = 16B

Header (no response, no encryption): 0x55, 0x01

Queue mouse reports (0x23)

Short description:	Puts HID mouse reports into buffer.	
CMD byte:	0x23	
Authentication required:	Yes	
Since FW version:	0.94	
PARAM:	Number of HID mouse reports in packet (1 - 32)	
DATA:	PARAM * HID mouse reports (Report1, Report2, ... ReportPARAM)	
RESP_CODE:	0x01	OK
	0x03	Buffer overflow (no reports were put into buffer!)
RESP_DATA:	None	
Details:	As long as you keep track of number of HID reports currently stored in buffer, there is no need to request response to this command. See HID chapter for mouse report details.	

Example:

Move scroll wheel by 5, move mouse pointer by X: 3, Y: 5, double left click (press left button → release left button → press left button → release left button)

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	HID Mouse Report 1				HID Mouse Report 2				HID Mouse Report 3				HID Mouse	
0x22	0x06	0x00	0x00	0x00	0x05	0x03	0x05	0x00	0x00	0x01	0x00	0x00	0x00	0x00	0x00

Report 4		HID Mouse Report 5				HID Mouse Report 5			
0x00	0x00	0x01	0x00	0x00	0x00	0x00	0x00	0x00	0x00

CRC32: 4B

Payload: 26B

Add 2B padding (0x00...)

Total length = 32B

Header (no response, no encryption): 0x55, 0x02

Queue gamepad reports (0x24)

Short description:	Puts HID gamepad reports into buffer.	
CMD byte:	0x24	
Authentication required:	Yes	
Since FW version:	0.97	
PARAM:	Number of HID gamepad reports in packet (1 - 32)	
DATA:	PARAM * HID gamepad reports (Report1, Report2, ... ReportPARAM)	
RESP_CODE:	0x01	OK
	0x03	Buffer overflow (no reports were put into buffer!)
RESP_DATA:	None	
Details:	<p>As long as you keep track of number of HID reports currently stored in buffer, there is no need to request response to this command.</p> <p>Gamepad report ID (1st report byte) is 0x03. See HID chapter for gamepad report details.</p>	

Example:

Gamepad buttons 1, 9, 10 are pressed, X axis = 0x20

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	HID Gamepad Report 1						
0x24	0x01	0x03	0x01	0x03	0x20	0x00	0x00	0x00

CRC32: 4B

Payload: 9B

Add 3B padding (0x00...)

Total length = 16B

Header (no response, no encryption): 0x55, 0x01

Queue mixed reports (0x25)

Short description:	Allows to send keyboard, mouse, consumer control and gamepad reports in a single packet	
<i>CMD</i> byte:	0x25	
Authentication required:	Yes	
Since FW version:	0.98	
<i>PARAM</i> :	Packet ID	
<i>DATA</i> :	Number of reports (max 33), Report 1 Type, Report 1 Payload,... Report N Type, Report N Payload, 0x00	
<i>RESP_CODE</i> :	0x01	OK
	0x05	Max reports exceeded (33 reports)
	0x06	Unknown report type
<i>RESP_DATA</i> :	None	
Details:	<p>Main reason for introducing this command was to allow to send several identical packets and be sure that only one will be executed. This allows to fix Android OS bug that under some conditions could corrupt data being sent to InputStick.</p> <p>Packet ID: 0x00 – skip ID check, accept packet other values – skip this packet if previous packet had the same Packet ID.</p> <p>Report types: 0x00 – end of payload 0x01 – HID keyboard report (standard) (length: 8B), 0x02 – HID keyboard report (short) (length: 2B), 0x03 – keyboard press and release event (length: 2B), 0x10 – HID mouse report (length: 4B), 0x20 – HID consumer report (length: 3B), 0x30 – HID gamepad report (length: 7B).</p> <p>If there is not enough free space to queue given packet into appropriate buffer, the packet is dropped.</p>	

Example:

Send 3 packets with identical HID reports: 1 keyboard press and release key event (press and release “a” key) and 2 mouse reports (press left mouse button, release left mouse button). Indicate end by adding end of payload report type (0x00). If first packet arrives, two following packets will be ignored.

Packet 1, Packet 2, Packet 3:

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	Report type	Report payload		Report type	Report payload				Report type	Report payload				Report type
0x25	0x01	0x03	0x00	0x04	0x10	0x01	0x00	0x00	0x00	0x10	0x00	0x00	0x00	0x00	0x00

CRC32: 4B

Payload: 16B

Add 12B padding (0x00...)

Total length = 32B

Header (no response, no encryption): 0x55, 0x02

Result: key “a” is pressed only once, left mouse button is pressed and released only once (even if all three packets will be correctly received).

When sending next set of packets, use different value of *PARAM*, for example 0x02. You can use following patterns for *PARAM*:

- 0x01, 0x02, ... 0xFF, 0x01, ...
- 0x01, 0x02, 0x01, 0x02, 0x01, ...

Write to endpoint (0x2B)

Short description:	Puts data into endpoint directly, without buffering.	
CMD byte:	0x2B	
Authentication required:	Yes	
Since FW version:	0.97	
PARAM:	Endpoint ID	
DATA:	LENGTH, report data (LENGTH bytes)	
RESP_CODE:	0x01	OK
	0x2A	Invalid endpoint ID
	0x2B	Endpoint busy (not empty), data was dropped.
	0x2C	Data exceeds size of endpoint
RESP_DATA:	None	
Details:	<p>This command allows to achieve minimal possible latency, however it is possible that data will be dropped. Should be used for gamepad or mouse functionality.</p> <p>Available endpoints (IDs): 0x01 – keyboard interface 0x02 – mouse interface 0x03 – consumer control (and gamepad) interface.</p> <p>Data will be put into endpoint only if it is empty at the moment the command is received. Otherwise data is dropped.</p>	

Example:

Put data into consumer control buffer (gamepad report: Gamepad buttons 1, 9, 10 are pressed, X axis = 0x20).

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	Endpoint ID	Length	Report data						
0x24	0x01	0x03	0x07	0x03	0x01	0x03	0x20	0x00	0x00	0x00

CRC32: 4B

Payload: 11B

Add 1B padding (0x00...)

Total length = 16B

Header (no response, no encryption): 0x55, 0x01

Queue SHORT keyboard reports (0x2C)

Short description:	Puts SHORT HID keyboard reports into buffer.	
CMD byte:	0x2C	
Authentication required:	Yes	
Since FW version:	0.97	
PARAM:	Number of „short” HID keyboard reports in packet (1 - 32)	
DATA:	PARAM * „short” HID keyboard reports (Report1, Report2, ... ReportPARAM	
RESP_CODE:	0x01	OK
	0x03	Buffer overflow (no reports were put into buffer!)
RESP_DATA:	None	
Details:	<p>As long as you keep track of number of HID reports currently stored in buffer, there is no need to request response to this command.</p> <p>See HID chapter for SHORT keyboard report details.</p> <p>This command is intended for Bluetooth Low Energy hardware version.</p>	

Example:

Use Ctrl + Alt + Delete key combination, release all keys, follow with ESC key, release ESC key.

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	SHORT Keyboard Report 1		SHORT Keyboard Report 2		SHORT Keyboard Report 3		SHORT Keyboard Report 4		SHORT Keyboard Report 5		SHORT Keyboard Report 6	
0x2C	0x06	0x05	0x00	0x05	0x4C	0x00	0x00	0x00	0x00	0x00	0x29	0x00	0x00

CRC32: 4B

Payload: 14B

Add 14B padding (0x00...)

Total length = 32B

Header (no response, no encryption): 0x55, 0x02

Queue Press and Release events (0x2D)

Short description:	Puts HID keyboard reports into buffer.	
CMD byte:	0x2D	
Authentication required:	Yes	
Since FW version:	0.97	
PARAM:	Number of „Press and Release” events in packet (1 - 10)	
DATA:	PARAM * „Press and Release” events (event1, event2, ... eventPARAM)	
RESP_CODE:	0x01	OK
	0x03	Buffer overflow (no reports were put into buffer!)
RESP_DATA:	None	
Details:	<p>Each „Press and Release” event is split into 3 HID keyboard reports before putting into buffer.</p> <p>As long as you keep track of number of HID reports currently stored in buffer, there is no need to request reponse to this command.</p> <p>See HID chapter for „Press and Release” event details.</p> <p>This command is intended for Bluetooth Low Energy hardware version.</p>	

Example:

Type „hEllO”, assume „en-US” compatible keyboard layout and CapsLock turned off.

CRC32			
0x??	0x??	0x??	0x??

CMD	PAR..	Event 1		Event 2		Event 3		Event 4		Event 5	
0x2D	0x05	0x00	0x0B	0x02	0x08	0x00	0x0F	0x00	0x0F	0x02	0x12

CRC32: 4B

Payload: 12B

Total length = 16B

(No need to add padding)

Header (no response, no encryption): 0x55, 0x01

2. InputStick → Application:

<i>CMD:</i>	Required:	FW >=
USB HID Status Update (0x2F)	Y	v0.94
Restore defaults Update (0x16)	N	v0.94
Error notification (0x1F)	N	v0.97

USB HID Status Update (0x2F)

Short description:	Provides status updates of USB interfaces and report buffers.	
<i>CMD</i> byte:	0x2F	
Since FW version:	0.94	
<i>DATA</i> :	Bytes:	Description:
	0	Device state
	1	Keyboard LEDs
	2	Keyboard protocol
	3	Keyboard buffer empty
	4	Mouse protocol
	5	Mouse buffer empty
	6	Consumer buffer empty
	7	Keyboard reports sent to host
	8	Mouse reports sent to host
	9	Consumer reports sent to host
	10	Reserved (0xFF)
Details:	This command will be sent by InputStick approximately every 100 ms.	

Keyboard LEDs:

Bit:	Description	Values
0	NumLock	0 – off, 1 - on
1	CapsLock	0 – off, 1 - on
2	ScrollLock	0 – off, 1 - on

Keyboard / Mouse protocol values:

Value:	Description:
0x00	report protocol
0x01	BOOT protocol

When mouse uses BOOT protocol, scroll wheel is not used. Keyboard is not affected by protocol value.

Keyboard / Mouse / Consumer buffer empty:

Value:	Description:
0x00	not empty

0x01	buffer is empty
------	-----------------

Keyboard / Mouse / Consumer reports sent to host:

such number of HID reports of given type has been successfully sent to USB host (since last update packet was sent) and was removed from respective report buffer.

Usage example: see (6) examples.pdf.

Restore defaults Update (0x16)

Short description:	Provides status updates of USB interfaces and report buffers.	
CMD byte:	0x16	
Since FW version:	0.94	
DATA:	Bytes:	Description:
	0	<i>RestoreStatus</i>
	1	<i>RestoreProgress</i>
	2	<i>TotalSteps</i>
	3	<i>NextLEDs</i>
	4	<i>TimeLeft</i>
Details:	<p>This command will be sent by InputStick only after „Restore defaults” procedure was initiated.</p> <p><i>RestoreStatus</i> – current status of restore defaults procedure. 0x00 – restore procedure is currently DISABLED, 0x01 - restore procedure is currently in progress (ENABLED). 0x02 – restore procedure was terminated due to keyboard LEDs not matching requested pattern (<i>NextLEDs</i>).</p> <p><i>RestoreProgress</i> – steps completed so far.</p> <p><i>TotalSteps</i> – total steps that must be completed.</p> <p><i>NextLEDs</i> – state of Keyboard LEDs, which must be set when <i>TimeLeft</i> reaches 0.</p> <p><i>TimeLeft</i> – time in seconds, until next check will be performed.</p> <p>When <i>TimeLeft</i> reaches 0, state of keyboard LEDs is checked. If it exactly matches <i>NextLEDs</i>, restore procedure will advance to a next step. If not it will be immediately disabled.</p> <p>When all steps are completed, InputStick will be restored to a factory default condition and enter infinite loop. It must be unplugged for USB port before it can be used again.</p> <p>Example: 0x01, 0x02, 0x0A, 0x03, 0x0E</p> <p>Restore procedure is currently in progress (ENABLED). Completed steps: 2 out of 10. During next check, keyboard LEDs must be set following way: NumLock ON, CapsLock ON, ScrollLock OFF. Remaining time until next check: 15s.</p>	

Error notification (0x1F)

Short description:	Provides status updates of USB interfaces and report buffers.	
<i>CMD</i> byte:	0x1F	
Since FW version:	0.97	
<i>DATA</i> :	Bytes:	Description:
	0 .. 10	<i>ERROR_CODE</i>
Details:	This command will be sent by InputStick to notify that some sort of error occurred. Contains up to 11 most recent <i>ERROR_CODES</i> . After sending this command, list of <i>ERROR_CODES</i> is cleared.	

ERROR_CODES:

Value:	Description:
0x00	Empty / no error
0xE0	USB transmission error
0xE1	Invalid <i>START_TAG</i>
0xE2	<i>CRC32</i> of recently received packet is different from 0x00000000 (ignore check) and incorrect. Packet was dropped.

~~BUG (v0.97): in some cases all data bytes will be filled with the same error code, even though the error occurred only once. (fixed in v0.98)~~

~~BUG (v0.97): 0xE2 will be returned even when *CRC32* was set to 0x00000000. Command will be accepted and executed anyway. (fixed in v0.98)~~