

# STICK-KU1/KE1 Remote Protocol

There are two ways to control the STICK, the first method uses a quick triggering message and the second more powerful method uses a button simulation.

## 1 – Quick Triggering

If you wish to trigger a specific scene without waiting for an answer from the STICK, use the datagram described below. All the datagrams must be sent via UDP protocol on port 2430.

UDP Datagram to trigger STICK's scenarios (12 Bytes) :

Field	Name	Size	Description
1	ID[8]	8 bytes	Array of 8 characters. Value must be "Stick_U1"
2	OpCode	2 bytes	Operation code. Value must be 6
3	Address	2 bytes	Scene number from 1 to 200. (8 buttons x 25 pages)

Examples:

For triggering scene number 2 (Page A, button 2):

Stick\_U1 6 002 → 0x53 0x75 0x6E 0x6C 0x69 0x74 0x65 0x5F 0x06 0x00 0x02 0x00

For triggering scene 20 (Page C, button 4):

Stick\_U1 6 020 → 0x53 0x75 0x6E 0x6C 0x69 0x74 0x65 0x5F 0x06 0x00 0x14 0x00

Note that each byte is inverted because of the endianness:

20 in decimal → 14 in hexadecimal → 00 14 as a Byte → 14 00 in the datagram.

## 2 – Button Simulation

UDP Datagram to trigger STICK's buttons (13 Bytes).

Field	Name	Size	Description
1	ID[8]	8 bytes	Array of 8 characters. Value must be "Stick_U1"
2	OpCode	2 bytes	Operation code. Value must be 101
3	SceneButton	Unsigned char	Scene number 1 → 1 Scene number 2 → 2 Scene number 3 → 4 Scene number 4 → 8 Scene number 5 → 16 Scene number 6 → 32 Scene number 7 → 64 Scene number 8 → 128
4	OtherButton	Unsigned char	Other buttons : Page Down → 1 Page Up → 2 Select → 4 Blackout → 8
5	SliderButton	Unsigned char	Represents the slider's value from 0 to 100

→ To build the !"#%&'(\$, you just need to understand that each bit of the char is linked to a button. For example:

If you want to simulate button 1, !"#%&'(\$ will be:

1 (in binary 00000001, in hexa 0x01)

If you want to simulate button 2, !"#%&'(\$ will be:

2 (in binary 00000010, in hexa 0x02)

If you want to simulate button 5, !"#%&'(\$ will be:

16 (in binary 00010000, in hexa 0x10)

If you want to simulate button 8, !"#%&'(\$ will be:

128 (in binary 10000000, in hexa 0x80)

→ To build )' \*#%&'(\$, it's the same. For example:

PageDown is 1 (0x01)

PageUp is 2 (0x02)

Select is 4 (0x04)

BlackOut is 8 (0x08)

→ The last char !, -, #%&'(\$ is just the slider value between 0 and 100.

Example in C:

```
int i = 85;
```

```
Byte b = (Byte) i;
```

Below are a few examples:

Empty datagram:

STICK\_U1 101 000 000 000 → 0x53 0x74 0x69 0x63 0x6B 0x5F 0x55 0x31 0x65 0x00 0x00 0x00 0x00

Triggering scene 5 datagram:

STICK\_U1 101 016 000 000 → 0x53 0x74 0x69 0x63 0x6B 0x5F 0x55 0x31 0x65 0x00 0x10 0x00 0x00

Pressing Select button datagram:

STICK\_U1 101 000 004 000 → 0x53 0x74 0x69 0x63 0x6B 0x5F 0x55 0x31 0x65 0x00 0x00 0x04 0x00

Changing slider value to 85 datagram:

STICK\_U1 101 000 000 085 → 0x53 0x74 0x69 0x63 0x6B 0x5F 0x55 0x31 0x65 0x00 0x00 0x00 0x55

All datagrams must be sent to the stick via UDP on port 2430.

Finally, you must send an empty datagram after each "button pressed" datagram so the stick understands that the "virtual finger has been lifted up from it. This allows you to simulate the multi-touch capabilities of the STICK in addition to stopping and pausing scenes. For example :

Activate audio triggering (Select + Button 7):

STICK\_U1 101 064 004 000 → 0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x65 0x00 0x40 0x04 0x00

Pressing buttons 5 and 8 simultaneously :

16 | 128 (in binary 10010000, in hexa 0x90)

STICK\_U1 101 144 000 000 → 0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x65 0x00 0x90 0x00 0x00

Stop Scene 8 :

STICK\_U1 101 128 000 000 → 0x53 0x74 0x69 0x63 0x6B 0x5F 0x55 0x31 0x65 0x00 0x80 0x00 0x00

Wait 4 seconds

STICK\_U1 101 000 000 000 → 0x53 0x74 0x69 0x63 0x6B 0x5F 0x55 0x31 0x65 0x00 0x00 0x00 0x00

# STICK-KU1/KE1 Remote Protocol Feedback

## Understanding STICK KE1/KU1 feedback messages

Each time the stick is touched or accessed remotely it broadcasts its new LED state on port 2430 using the UDP protocol.

This datagram is always 14bytes long.

Stick\_U1(8bytes) Op Code(2bytes) Mode(2bytes) State(2bytes)

Op Code(2bytes) → Allows you to know which LEDs are affected by the datagram (scenes, pages, mode or slider).

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x68 0x0 0x0 0x0 0x19 0x0

**MODE BUTTON (Op Code = 0x69):**

**Example:** Mode Dimmer:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x69 0x0 0x0 0x0 0x0 0x0

**Example:** Mode Speed:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x69 0x0 0x0 0x0 0x01 0x0

**Example:** Mode Color:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x69 0x0 0x0 0x0 0x02 0x0

**SLIDER (Op Code = 0x6a):**

The slider value is from 0 to 1000.

**Example:** Slider at 0:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x6a 0x0 0x01/0x02 0x0 0x0 0x0

**Example:** Slider at 50%:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x6a 0x0 0x01/0x02 0x0 0xf4 0x01

$0xf4 + (256 * 0x01) = 254 + (256 * 1) = 500$

**BLACKOUT BUTTON(Op Code = 0x66/0x6b):**

**Example:** BLACKOUT enabled:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x66 0x0 0x0 0x0 0x1 0x0

**Example:** ALL LEDs OFF:

0x53 0x74 0x69 0x63 0x6b 0x5f 0x55 0x31 0x6b 0x0 0x0 0x0 0x0 0x0

NOTE: Although this is an official release of the protocol, all the commands described are subject to change without notice. Therefore please check with us before performing a firmware upgrade.