DJ Link Packet Analysis

James Elliott
Deep Symmetry, LLC

April 26, 2016

Abstract

The protocol used by Pioneer professional DJ equipment to communicate and coordinate performances can be monitored to provide useful information for synchronizing other software, such as light shows and sequencers. By creating a "virtual CDJ" that sends appropriate packets to the network, other devices can be induced to send packets containing even more useful information about their state. This paper documents what has been learned so far about the protocol, and how to accomplish these tasks.

1 Mixer Startup

When the mixer starts up, after it obtains an IP address (or gives up on doing that and self-assigns an address), it sends out what look like a series of packets[1] simply announcing its existence to UDP port 50000 on the broadcast address of the local network.

These have a data length of 37^2 bytes, appear roughly every 300 milliseconds, and have the content shown in Figure 1.

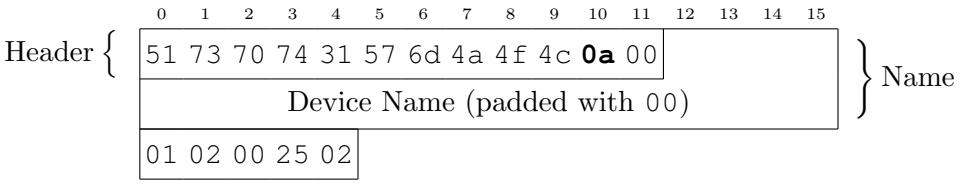

Figure 1: Initial announcement packets from Mixer

[1]The packet capture described in this analysis can be found at `https://github.com/brunchboy/dysentery/raw/master/doc/assets/powerup.pcapng`

[2]Values within packets are shown in hexadecimal, while packet lengths and byte offsets are discussed in decimal.

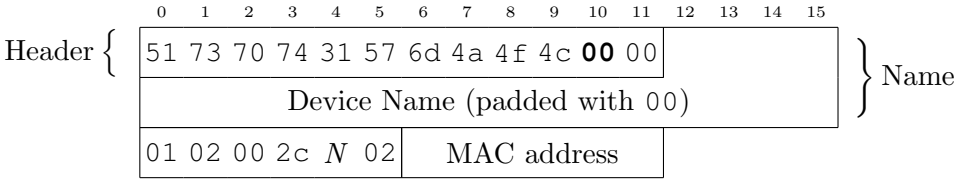

Figure 2: First-stage Mixer device number assignment packets

The tenth byte (inside what is labeled the header) is bolded because its value changes in the different types of packets which follow.

After about three of these packets are sent, another series of three begins. It is not clear what purpose these packets serve, because they are not yet asserting ownership of any device number; perhaps they are used when CDJs are powering up as part of the mechanism the mixer can use to tell them which device number to use based on which network port they are connected to?

In any case, these three packets have a data length of 44 bytes, are again sent to UDP port 50000 on the local network broadcast address, at roughly 300 millisecond intervals, and have the content shown in Figure 2.

The value N at byte 36 is 1, 2, or 3, depending on whether this is the first, second, or third time the packet is sent.

After these comes another series of three numbered packets. These appear to be claiming the device number for a particular device, as well as announcing the IP address at which it can be found. They have a data length of 50 bytes, and are again sent to UDP port 50000 on the local network broadcast address, at roughly 300 millisecond intervals, with the content shown in Figure 3.

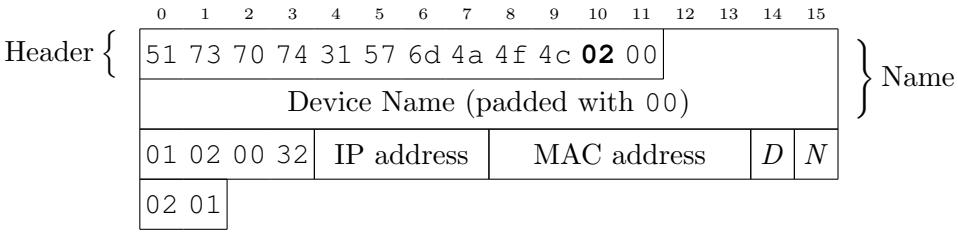

Figure 3: Second-stage Mixer device number assignment packets

I identify these as claiming/identifying the device number because the value D at byte 46 is the same as the device number that the mixer uses to identify itself (`0x21`) and the same is true for the corresponding

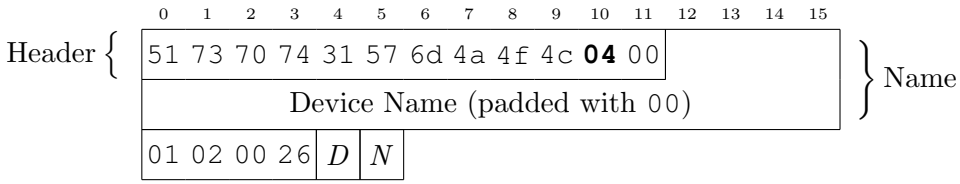

Figure 4: Final-stage Mixer device number assignment packets

packets seen from the CDJs (they use device numbers 2 and 3, as they are connected to those ports/channels on the mixer).

As with the previous series of three packets, the value N at byte 47 takes on the values 1, 2, and 3 in the three packets.

These are followed by another three packets, perhaps the last stage of claiming the device number, again at 300 millisecond intervals, to the same port 50000. These shorter packets have 38 bytes of data and the content shown in Figure 4.

As before the value D at byte 36 is the same as the device number that the mixer uses to identify itself (`0x21`) and N at byte 37 takes on the values 1, 2, and 3 in the three packets.

Once those are sent, the mixer seems to settle down and send what looks like a keep-alive packet to retain presence on the network and ownership of its device number, at a less frequent interval. These packets are 54 bytes long, again sent to port 50000 on the local network broadcast address, roughly every second and a half. They have the content shown in Figure 5.

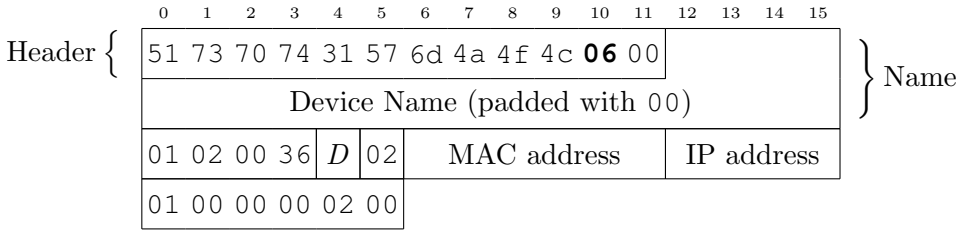

Figure 5: Mixer keep-alive packets

2 CDJ Startup

When a CDJ starts up the procedure and packets are nearly identical, with groups of three packets sent at 300 millisecond intervals to port 50000 of the local network broadcast address. The only difference

between Figure 6 and Figure 1 is the final byte, which is `0x01` for the CDJ, and was `0x02` for the mixer.

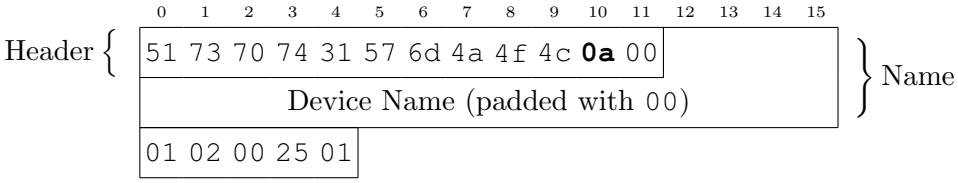

Figure 6: Initial announcement packets from CDJ

Similarly, the next series of three packets from the CDJ are nearly identical to those from the mixer. The only difference between Figure 7 and Figure 2 is byte 37 (immediately after the packet counter N), which again is `0x01` for the CDJ, and was `0x02` for the mixer.

However it appears that in this capture the CDJ skips the second stage of claiming a device number, probably because it is configured to be automatically assigned a device number based on the port of the mixer to which it is connected, and we cannot see a packet that the mixer sent it assigning it that device number. Instead, it jumps right to the end of the third and final stage, sending a single 38-byte packet with header byte 10 set to tt 04 (which identified the three packets of the third stage when the mixer was starting up), with content identical to Figure 4.

Even though the value of N is `01`, this is the only packet in this series that the CDJ sends. It would probably behave differently if configured to assign its own device number (behaving like we saw the mixer behave in claiming its device number).

The CDJ then moves to the keep-alive stage, sending out 54-byte packets with the content shown in Figure 8.

As seems to always be the case when comparing mixer and CDJ packets, the difference between this and Figure 5 is that byte 37 (following the device number D) has the value `01` rather than `02`, and the same is true of the second-to-last byte in each of the packets. (Byte

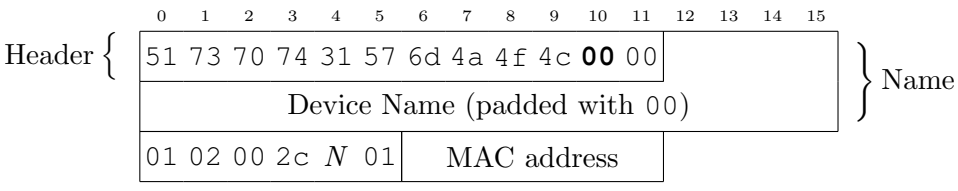

Figure 7: First-stage CDJ device number assignment packets

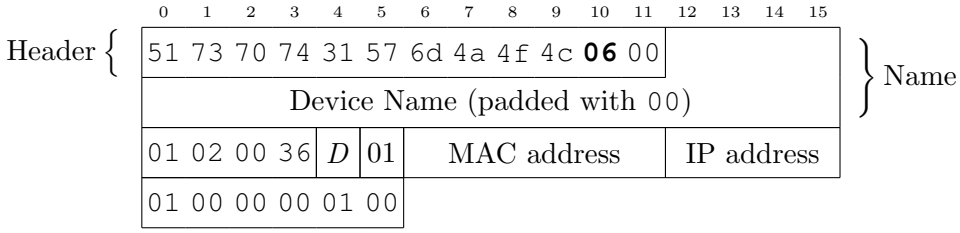

Figure 8: CDJ keep-alive packets

52 is `01` in Figure 8 and `02` in Figure 5.

3 Tracking BPM and Beats

For some time now, Afterglow[3] has been able to synchronize its light shows with music being played on Pioneer equipment by observing packets broadcast by the mixer to port 50001. Until recently, however, it was not possible to tell which player was the Master, so there was no way to determine the down beat (the start of each measure). This section will be expanded and more details provided as Afterglow is updated to take advantage of the discoveries described in the next section.

Until then, here is a summary of what is currently done. A socket is opened and bound to port 50001. Whenever a packet from the mixer is received on this socket, if the length is 96 bytes, it is known to contain beat and BPM information. The current BPM can be obtained as:

$$\frac{byte[90] \times 256 + byte[91]}{100}$$

These packets are sent on each beat, and the current beat number (1, 2, 3 or 4) is sent in *byte*[92]. However, the beat number is *not* synchronized with the master player, and so it is not useful for much. We expect to make use of the Virtual CDJ technique to determine the actual beat number soon.

4 Creating a Virtual CDJ

Although some useful information can be obtained simply by watching broadcast traffic on a network containing Pioneer gear, in order to

[3] `https://github.com/brunchboy/afterglow#afterglow`

get important details it is necessary to cause the gear to send you information directly. This can be done by simulating a "Virtual CDJ".[4]

To do this, bind a UDP server socket to port 50002 on the network interface on which you are receiving DJ-Link traffic, and start sending keep-alive packets to port 50000 on the broadcast address as if you were a CDJ. Follow the structure shown in Figure 8, but use the actual MAC and IP addresses of the network interface on which you are receiving DJ-Link traffic, so the devices can see how to reach you.

You can use a value like 5 for *D* (the device/player number) so as not to conflict with any actual players you have on the network, and any name you would like. As long as you are sending these packets roughly every 1.5 seconds, the other players and mixers will begin sending packets directly to the socket you have opened on port 50002.

We are just beginning to analyze all the information which can be gleaned from these packets, but here is what we know so far.[5]

4.1 Mixer Status Packets

Packets from the mixer will have a length of 56 bytes and the content shown in Figure 9.

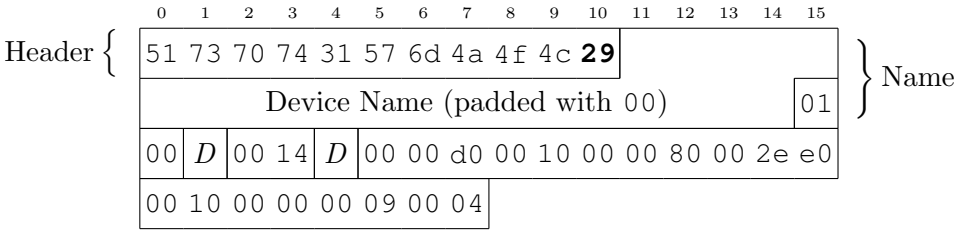

Figure 9: Mixer status packets

Packets coming from a DJM-2000 nexus connected as the only mixer on the network contain a value of 33 (`0x21`) for their Device Number *D* (bytes 33 and 36).

4.2 CDJ Status Packets

Packets from a CDJ will have a length of 212 bytes and the content shown in Figure 10.

[4]Thanks are due to Diogo Santos for discovering the trick of creating a virtual CDJ in order to receive detailed status information from other devices.

[5]Examples of packets discussed in this section can be found in the capture at `https://github.com/brunchboy/dysentery/raw/master/doc/assets/to-virtual.pcapng`

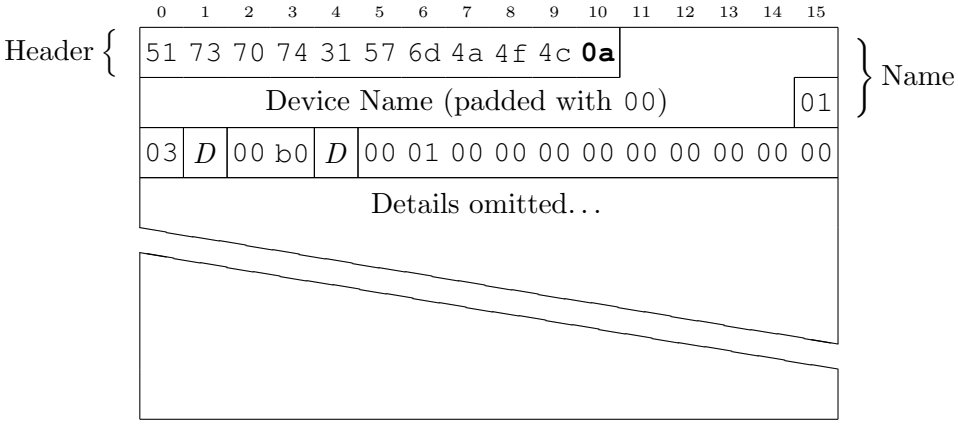

Figure 10: CDJ status packets (partial)

This is only a partial diagram, and will be fleshed out as we progress with our analysis. The Device Number in *D* (bytes 33 and 36) is the Player Number as displayed on the CDJ itself. In the case of this capture, the CDJs were assigned Player Numbers 2 and 3.

We do currently know that byte 137 is a bit field containing some very useful state flags, detailed in Figure 11.

7	6	5	4	3	2	1	0
1	Play	Master	Sync	On-Air	1	0	0

Figure 11: CDJ state flag bits

We have not yet seen any other values for bits 0–2 or 7, so are unsure if they also carry meaning. If you ever find different values for them, please let us know by filing an Issue! `https://github.com/brunchboy/dysentery/issues`