IDN-NetConfig: A Flexible Extension of the IDN-Hello Protocol (ILDA Digital Network) to Control Application Specific Parameters

Matthias Frank
University of Bonn
Institute of Computer Science 4 / Laser & Light Lab
Friedrich-Hirzebruch-Allee 8, D-53115 Bonn, Germany
matthew@cs.uni-bonn.de
http://lll.net.cs.uni-bonn.de/

***Abstract* —The ILDA Digital Network (IDN) is a novel protocol family providing digital data transmission for laser projection. While the new standards at first glance aim to replace the old analogue signal transmission, the digital streaming concept also enables completely new applications and flexible networked scenarios. The IDN-Hello protocol is the basic protocol for exchange of information between IDN enabled devices. It also includes procedures for IDN device and service discovery as well as exchange of laser specific parameters. Our paper presents an extension to IDN-Hello to allow for a configuration of application specific parameters via the local network, our so-called "IDN-NetConfig". The virtual demo will showcase IDN-NetConfig with two applications for laser show data visualization.**

I. Introduction to IDN & Demo Motivation

Laser shows are widely used for entertainment purposes, e.g. to accompany music performances. A typical laser projector consists of a laser module and a fast-moving mirror system, which deflects the laser beam to create shapes and patterns either on a projection surface or in mid-air when using artificial haze or fog. More on laser show basics in [1] and [2].

The International Laser Display Association (ILDA) defines the de facto technical standards used in both laser show hardware and software. Traditionally, laser projectors are controlled with analogue input signals. The analogue ILDA Standard Projector (ISP, [3], the 25-pin connector is also called ISP-DB25) was the relevant specification on how to connect laser control interfaces with laser projectors in a vendor-independent way, which is in use for more than two decades (technical specification of 1999).

The ILDA Digital Network (IDN) Stream Specification is a novel standard to transmit laser show artwork in digital data streams inside a computer network [4]. The digital processing allows for very flexible setups as opposed to the old analogue interfacing and cabling systems. For example, in previous work we have already shown how an IDN software driver can complement existing laser show software to directly generate IDN streams [5]. These are being sent into the local network with destination to an IDN capable laser projector or a suitable IDN-to-analogue converter for legacy ILDA/ISP-DB25 controlled laser projectors.

Furthermore, the transmitted IDN streams can easily be received and analyzed to help with testing and debugging of laser show systems. The IDN-Toolbox is able to display the received IDN stream(s) on a 2D computer screen [6]. With the IDN-Laser-VR software, a more authentic approach on laser show visualization from IDN streams using virtual reality (VR) headsets is possible: Using a VR headset, the user can move around in a virtual room with several virtual laser projectors and watch laser show artwork generated by typical laser show software. Even without a VR headset connected, the user will see a 3D preview on the computer screen [7].

Another IDN specification is still work in progress: The IDN Discovery Protocol (aka IDN-Hello) describes the protocol for the detection, enumeration and query of IDN-enabled devices and currently also is the basis for the IDN Stream protocol encoding and sending IDN laser projection data over the network. Current IDN hardware and all of our IDN tools already support IDN-Hello unit and service discovery in combination with IDN Stream for laser data transport.

Furthermore, the IDN-Hello protocol will have a mechanism and respective commands (request, response) to allow IDN network elements (sender, receiver) to exchange laser specific parameters. In the context of developing our IDN visualization software (IDN-Toolbox, IDN-Laser-VR) and lastly making them capable to be compliant to the IDN-Hello service discovery, the idea was born to propose an extension of the IDN-Hello protocol to allow to also control application specific parameters of IDN receivers in a yet flexible, but independent way of laser specific parameters that otherwise would need to be fixed in a stable or final version of the IDN-Hello protocol specification.

The next section of this demo paper presents all necessary information about our proposed extension, the "IDN-NetConfig". Section III gives an outline of what to see at the virtual demo and concludes the paper.

II. Subject of the Demo: IDN-NetConfig extionsion of IDN-Hello

Figure 1 shows a simplified architecture of a local IDN network: The so-called IDN producers are depicted in the top

left corner, these are generating one or more IDN streams and are sending this data via the local network to one or more IDN consumers (receivers of the IDN stream(s)). Typically, the IDN producer is a laser show software or some tool that is directly generating the IDN stream data. Figure 1 shows several IDN consumers on the right: An IDN converter that could be attached to or integrated into a laser projector, and also the visualization software tools IDN-Toolbox and IDN-Laser-VR.

With the IDN-Hello protocol, every IDN packet that is being transmitted in the network starts with a 4-byte packet header, carrying a command byte, a flags byte and a 16-bit sequence number. All commands referring to IDN Stream and real-time laser data transmission are using a value of 0x40 (hexadecimal) or subsequent (as of the current draft specification 0x41, 0x44, 0x45, 0x47).

The commands for unit discovery are 0x10 scanning the network for IDN units using a local broadcast and 0x11 for the (unicast) response with unit identification and status. As a follow-up step with command 0x12 an IDN producer can ask the IDN consumer for available IDN services. This is a separate step as a single IDN consumer unit can offer several IDN services. The reply is command 0x13 with a map of supported services, which has a dynamic size consisting of several service map entries.

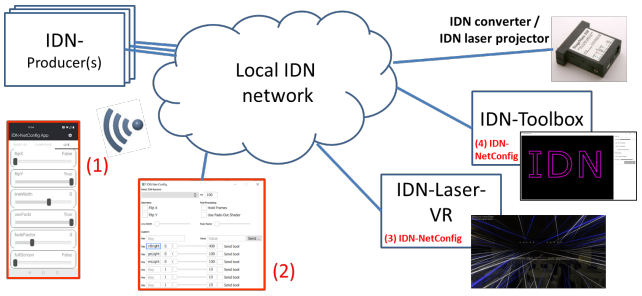

Figure 1. **IDN network architecture, incl. IDN-NetConfig elements.**

For the exchange of laser specific parameters, currently the command values 0x20 to 0x29 are reserved, allowing for request and response of service, unit or link specific parameters. Details for the concept and procedures are still to be finalized (and are not yet implemented in existing IDN producer/consumer software or hardware).

For our proposed extension for control of application specific parameters we are using specific new command values of the IDN-Hello packet header. It has been negotiated that the IDN-Hello specification will reserve command values 0xC0 and higher (up to 0xFF) for vendor specific use cases.

Figure 1 also shows our specific use cases: The IDN-Toolbox (red mark "(4)…" and IDN-Laser-VR (red mark "(3)…") are individual examples for IDN consumer software that can benefit from the capability of controlling specific application parameters remotely via the network instead of controlling these parameters in the instance of the application itself (e.g. via an appropriate graphical user interface, GUI). The red marks "(1)" and "(2)" in Figure 1 are placeholders for an Android App and a Qt based software with GUIs to control those application specific parameters independently of the IDN producers of the IDN laser streams.

The vendor specific IDN-Hello command 0xF1 with a local broadcast (same as for unit scan 0x10) is used to scan the network for IDN-NetConfig enabled units. Only these will respond with the command 0xF2. Now those units are able to receive command 0xF0, which is used to set a specific parameter of the application to a certain value.

For ease of implementation and debugging, the 0xF0 command contains the usual 4-byte IDN-Hello packet header, a parameter keyword (as 16-byte string) and the parameter value (as 20-byte string). This looks inefficient in terms of number of bytes used in the payload, but as the frequency of parameter changes is dominated by actions of a human user using a software/GUI like (1) or (2) (cf. Figure 1), the total amount of updates/changes per time interval and thus the generated amount of network traffic is negligible.

Sender and receiver software can make use of prominent programming functions of formatted printing to and reading from strings (`sprintf()` and `sscanf()`), the definition and use of keywords is not limited at all (just the length of 16 chars) and it is easy to debug the payload content with network analysis tools like Wireshark.

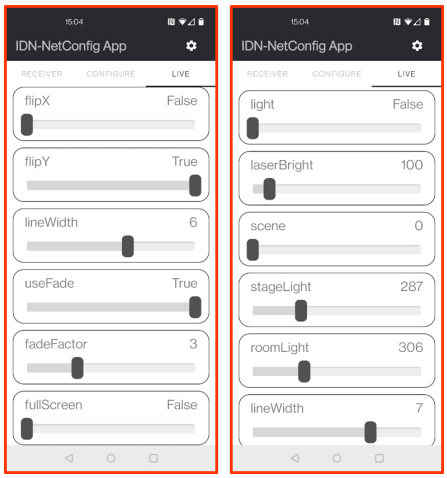

Figure 2. **Android App for the IDN-NetConfig controller, (left) for IDN-Toolbox, (right) for IDN-Laser-VR.**

Figure 2 shows screenshots of the Android IDN-NetConfig controller App with typical application specific parameters for IDN-Toolbox (left) and IDN-Laser-VR (right). E.g. the parameter "lineWidth" is used in both applications and controls the thickness of lines for visualizing the laser graphic drawing in the visualization software. Other parameters are unique for each application. Due to space limitations of this paper, these parameters will be showcased and explained in more detail only in the virtual demo.

The IDN-NetConfig Android App is designed to be fully configurable concerning the keyword strings and allows to use Boolean type (values true and false) or unsigned integer with a selectable value range. For both types, the GUI uniformly uses a slider to allow the user to change the parameter values. Figure

3 (right) shows the Android App in configuration mode: An arbitrary number of parameters with type and value range maybe configured, the order of appearance maybe changed and a parameter preset maybe stored, e.g. Figure 2 (left) vs. (right).

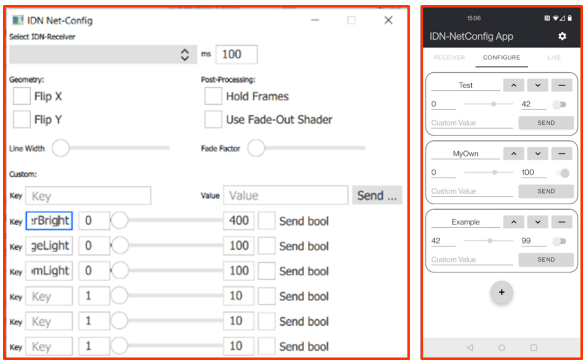

Figure 3. **Flexible selection of arbitrary parameters, (left) in Qt GUI, (right) in Android App.**

Figure 3 (left) shows the GUI of a Qt based version of the IDN-NetConfig controller. The major set of parameters of the IDN-Toolbox is hard-coded with GUI elements like checkboxes (Boolean) and sliders (e.g. again "Line Width"). The bottom part shows six sliders that can be used to control arbitrary parameters (Boolean; integer with selectable value range) by typing the appropriate keyword string into the "Key" text fields.

The receiving side of IDN-NetConfig as a software library within the IDN consumer application is storing all incoming keyword – value pairs in a dynamic list and is remembering only the latest value for each parameter. The specific application can query the IDN-NetConfig library and can look up the value string for a specific parameter keyword.

III. What to See at the Demo & Conclusion

In the virtual demo the participants will be able to see a set of IDN consumer applications with IDN-NetConfig extension from the Laser & Light Lab of the University of Bonn, via screen sharing, camera scene composition or a combination of both. A suitable IDN producer will create IDN streams with laser projection data in a loop, while the IDN-NetConfig controller software (both Qt based and Android App) will be showcased to change those application specific parameters of the IDN-Toolbox and the IDN-Laser-VR. The best motivation for network based control of these parameters will be visible with the IDN-Toolbox in the so called full-screen mode, where the software has no own GUI elements for control of any parameters and is able to be operated on a computer even without human interface devices as keyboard or mouse.

To conclude, in general the IDN-NetConfig concept can be considered as a blue print for other application specific use cases with IDN consumers. The IDN-Toolbox and IDN-Laser-VR use a common software library for decoding the content of IDN stream messages that is also implementing the receiving side of IDN-NetConfig. This library is able to be re-used by other software, using their own application specific parameters and polling values from the library with appropriate keyword strings. The IDN-NetConfig controller software in both Qt version and in particular Android App are already flexible to configure individual keyword strings and select Boolean or signed integer type.

Currently, our IDN-NetConfig implementation only supports unicast transmission of keyword – value pairs towards the IDN consumer (command 0xF0). This could be extended by also requesting feedback from the IDN consumer (request response cycle) or even enabling the IDN consumer itself to request some information from the IDN-NetConfig controller source. A possible use case for this might be a mutual authentication of IDN producer and consumer before starting IDN stream transmission, activation of application specific encryption of IDN stream content with prior key exchange, or similar handshaking or configuration purposes that are independent of the laser specific encoding of data and other laser specific parameters.

It is planned to publish a video of the IDN-NetConfig demonstration in the YouTube channel of the Laser & Light Lab of Uni Bonn [8] and to release the existing software of our current IDN-NetConfig implementation in a public gitlab repository [9].

Acknowledgement: The work on the IDN-NetConfig concept started in a Computer Science Bachelor project course in joint activity by Georg Kuhlemann und Sebastian Tasch in 2020. In their Bachelor Theses, Georg completed the IDN-NetConfig elements in Qt and within the IDN-Toolbox, and Sebastian developed the Android App und provided IDN-NetConfig support in the IDN-Laser-VR software, all in 2021.

References

[1] Laser F/X International. How Laser Shows Work - Introduction, https://laserfx.com/Works/IndexWorks.html (last accessed Sept. 28, 2021)

[2] Michael Roberts, Richard Gonsalves. Laser F/X: The Light Show Handbook (Mark II), R.A.G.e. Media, July 2021

[3] ILDA, ILDA Technical Standards – International Laser Display Association, Orlando/Florida, USA, https://www.ilda.com/technical.htm (last accessed Sept. 28, 2021)

[4] Matthias Frank, Horacio Pugliese, Andrew Berry, Tim Walsh, and Dirk Apitz. The ILDA Digital Network Stream Specification. International Laser Display Association, Tech. Rep. Revision 001, 2015

[5] Matthias Frank, A Multi-Platform Library for a Software Sender for the (proposed) ILDA Digital Network, Demo/International Conference on Networked Systems, NetSys 2015, Cottbus, March 2015

[6] Matthias Frank. Demonstration of "IDN-Toolbox": A Software to Visualize and Analyze IDN (ILDA Digital Network) Streams. Demonstrations of the 42nd IEEE Conference on Local Computer Networks (LCN), 2017

[7] Matthias Frank, Fabian Marquardt. IDN-Laser-VR: A Demonstration of Real-Time Multimedia Streaming in Local Networks Using the ILDA Digital Network Protocol. Demonstrations of the 44th IEEE Conference on Local Computer Networks (LCN), 2019

[8] Laser & Light Lab University of Bonn YouTube channel https://www.youtube.com/channel/UCHTP9mmi3p3Yri4faX8zyTw (last accessed Sept. 28, 2021)

[9] Laser & Light Lab University of Bonn gitlab Repository. Several IDN related projects. https://gitlab.com/laser_light_lab_uni_bonn/ (last accessed Sept. 28, 2021)