HBK
HOTTINGER BRÜEL & KJÆR

Programmer's Intro to Open API for LAN-XI

HBM

Brüel & Kjær

A Brüel & Kjær LAN-XI module offers great flexibility. At one end of the module, exchangeable front panels connect to a large variety of analog and digital signal connectors and supply transducers with the necessary current or voltage. At the other end of the module, an RJ45 Ethernet connector interfaces to your PC (typically) and to other LAN-XI modules, which can be distributed over a large area. The Ethernet cables handle not only the data in/out, but also supply power and synchronization at sample-level accuracy, allowing for, for example, cross-spectra between any two channels in a system.

In-between the signal connectors and the network connector, you get high quality signal conditioning and conversion from analog to digital (and vice-versa on some LAN-XI modules).

Distributing the LAN-XI modules typically saves a lot of expensive and heavy analog cables when measuring on large objects, effectively reducing ground-loop problems.

If you prefer to concentrate several measurement channels in one place, LAN-XI modules fit into frames that can be stacked or mounted in racks.

WHY USE THE OPEN API FOR LAN-XI?

Most of our customers are very happy with the solutions offered via the BK Connect® software platform, while some still prefer the toolbox-oriented PULSE™ LabShop. These Microsoft® Windows®-based software clients provide an interface to our LAN-XI modules, providing a large selection of analysis algorithms, as well as graphic displays with advanced cursors.

Figure 1. LAN-XI Bridge Input Module with three channels, 0 – 100 kHz measurement bandwidth.

Figure 2. LAN-XI modules in frames.

But what if you already have your own software, and you just want to use Brüel & Kjær LAN-XI modules as front ends for this? What if you have invested thousands of hours in your own software? What if your software also interfaces to other specific hardware, or your company database?

And what if you are developing your own software on Linux or Mac?

The Open API for LAN-XI could be the right solution for all the above scenarios.

Controlling a single LAN-XI module is relatively simple and should be your starting point. The Open API, however, also supports multiple sample-synchronous LAN-XI modules – in frames and/or distributed via Ethernet. It is possible to connect modules to a wireless router and then control them from a phone or similar, but we do recommend that you start with a wired solution, which is easier to debug.

TWO PROTOCOLS

The Open API is built on two protocols – one for setup and control and one for streaming. There are scenarios where you only need the setup protocol, but you cannot stream data without the setup.

The control-part of Open API is a wire protocol on top of http. This means that any modern operating system and language can be used. At github.com/hbk-world you can find sample code written in C# and Python. You can also download the latest firmware for LAN-XI here, as well as a draft of the next official version of the programmer's reference.

Since the setup protocol is based on http, the necessary libraries come with your favourite programming language. Another advantage is that most programmers already understand the basic 'idioms'. See table below, which also shows the similarity to the CRUD idioms in database programming.

HTTP Method	CRUD (database)	Description
Post	Create	Create a new resource (at parent-to-be) – or perform action
Get	Retrive	Retrieve a representation of a resource without side effects
Put	Update	Update a resource
Delete	Delete	Delete a resource

Streaming data from (or to) a LAN-XI module is a bit more complicated. The samples on GitHub® demonstrate – in C# and Python – how to interpret the binary stream. If you prefer another programming language, I suggest that you start with the Kaitai tool from github.com/hbk-world, and generate a parser for the streaming format, in your chosen programming language.

The streaming part of the Open API – and thereby part of the samples – requires a unique license installed in each of the LAN-XI modules used. This must be bought from HBK. Still, if you have a LAN-XI module without an Open API license, you can set it up to stream data to an SD card, perform a measurement and then retrieve it for viewing or data analysis.

GETTING STARTED WITH THE LAN-XI MODULE HOMEPAGE

In the following pages we will use two very nice third party tools:

- Wireshark – see https://www.wireshark.org/
  We use Wireshark to monitor the traffic on the cable. This is indispensable when debugging.
- Postman – see https://www.postman.com/
  We use Postman to generate test commands. An alternative is cURL.

LAN-XI modules have a homepage. An example is shown in Figure 3.

Like the setup protocol, the homepage is (by definition) based on http. This makes it a good place to start understanding the REST protocol, while digging into the functionality of the support we get from the homepage.

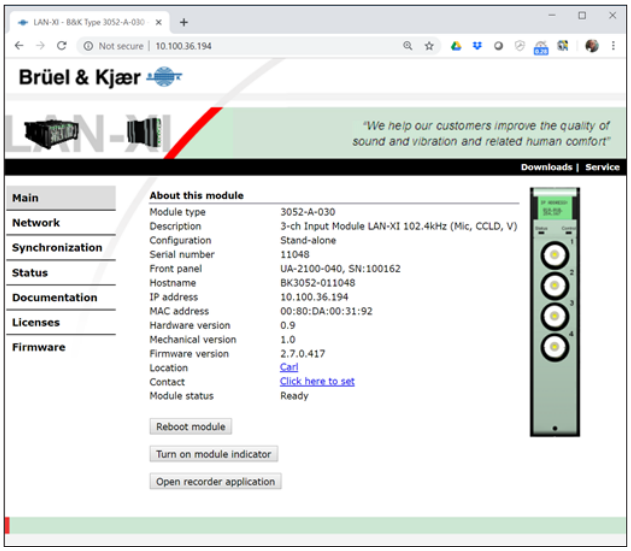

Figure 3. LAN-XI homepage

The LAN-XI module homepage provides a lot of relevant information:

- 1. The IP address – seen at the top in the browser address bar – here 10.100.36.194. It is also found on the page itself.
- 2. Serial numbers, front panel image, etc.
- 3. Clicking the **Synchronization** menu will give you information about the PTP-status.
- 4. Clicking the Licenses menu will take you to a page where you can paste in licenses that you might have received by, for example, email.
- 5. Clicking the **Firmware** menu will allow you to update firmware. There is also a PC application – PULSE Frontend-Setup – that can update a full system in one operation.
- 6. The button **Open recorder application** starts a web-application known as Notar™. This is the original foundation for the Open API. You can try it out and even use Wireshark to see what goes on behind the scenes.

Figure 4 shows how the retrieval of the above webpage looks in Wireshark. The top third of Wireshark shows each 'packet' as a single line. Here we have selected number 34, which is then decoded in the window below. Often there is a third window below this, which shows the same in hexadecimal. It is not shown here.

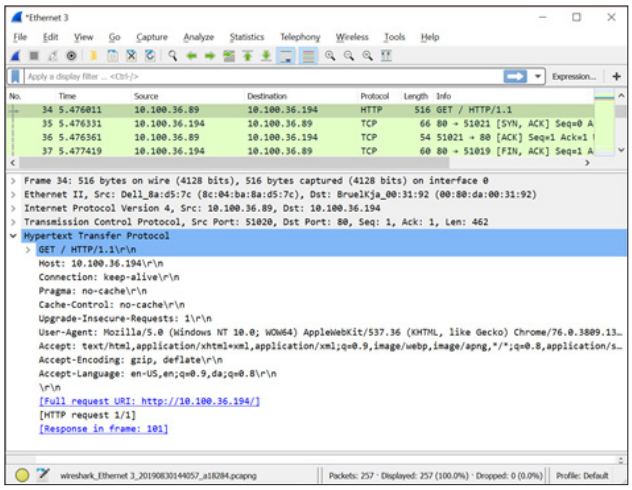

Figure 4. Wireshark showing GET of a homepage.

When you right-click on packet 34 in the top window and select Follow TCP Stream, you get the dialogue shown in Figure 5. This shows ALL the packets in the http conversation starting with packet 34. The HTTP request is shown in red, while the response is shown in blue.

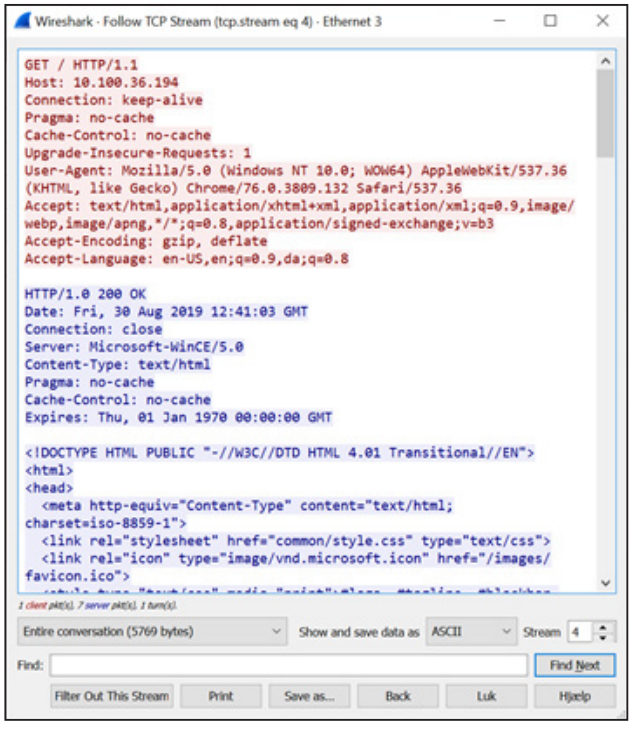

Figure 5. Request and response of a single HTTP command.

Without going into too much detail, we see that http messages contain:

- 1. Two mandatory lines. In the request, the first line contains the command GET; the HTTP standard used: 1.1; and the URL of the command: / ; while in the response we see the error code: 200 OK. The second mandatory line of the request contains the host part of the URL – in this case given as an IP-address.
- 2. Several optional lines follow – controlling 'pipelining', etc.
- 3. A double line break – \r\n\r\n – marks the end of the header and the start of the body.
- 4. A GET request has no body, but the GET response does – often with a lot of data (here HTML).

HOW TO SEND AND DEBUG REST COMMANDS

Instead of using the browser to retrieve webpages containing HTML, we can use it to send a simple command to get the information in JSON format from a LAN-XI Module. In the address bar of our browser we write the following (for same module):
http://10.100.36.194/rest/rec/module/info/

The first part of this URL is the IP-address, which obviously must match the module's address. Followed by /rest/rec, which must always be there to access the LAN-XI Open API. After this, we move into the module's hierarchical object model. In this case we access the module object and underneath this, the info object. Figure 6 shows the conversation in Wireshark.

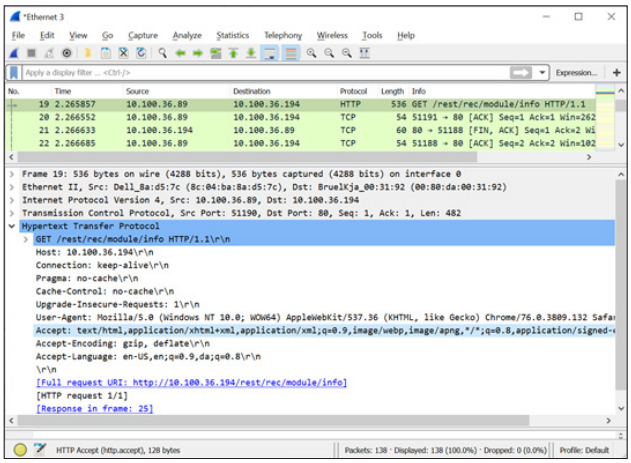

Figure 6. Wireshark shows the request for the module/info object.

Again, we right-click the start of the http-conversation (now packet 19) and select Follow TCP stream. The resulting dialogue is found in Figure 7. Note that the response this time says that the Content-Type is application/json – with a Content-Length of 1423. This makes it relatively easy for us to parse the JSON in our code – providing us with the relevant setup information.

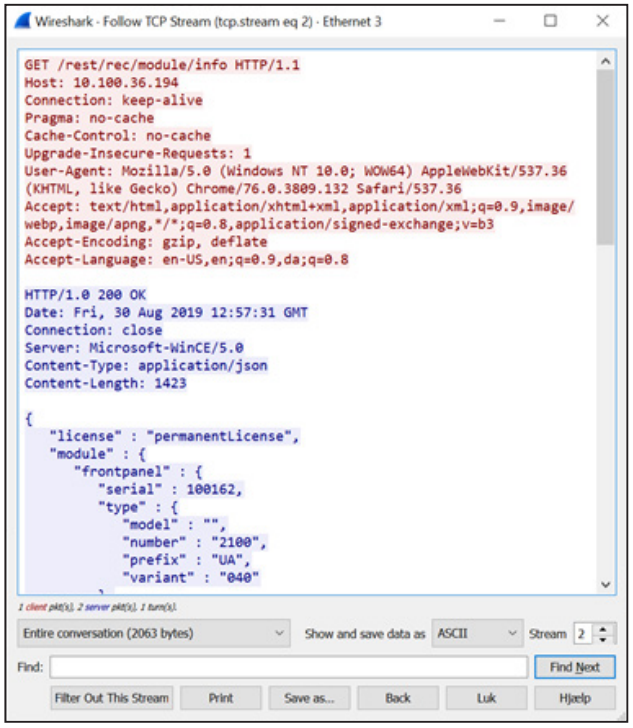

Figure 7. http conversation with JSON data.

Using the browser's address bar for writing GET commands is fine, because that is all an address bar does. It implicitly always sends GET commands with the given URL and these do not contain a message body.

However, when you need to send PUT commands, the browser cannot help you that much. This might be where you start writing your program, but if you like to handcraft your tests a little bit further, you can use Postman, which is a third-party program, with a large community. Figure 8 shows a PUT command sent via Postman. Note that PUT is selected in the drop-down box next to the IP address of the module. Postman automatically creates good default headers. Below this, we have written four lines of JSON that Postman will use in the body of the request. Further below we have the response – here we have chosen to look at the headers, since in this case the message body is empty.

Postman is clever. You can, for example, set the IP address as a global variable, so that you can save scripts that could work on any module.

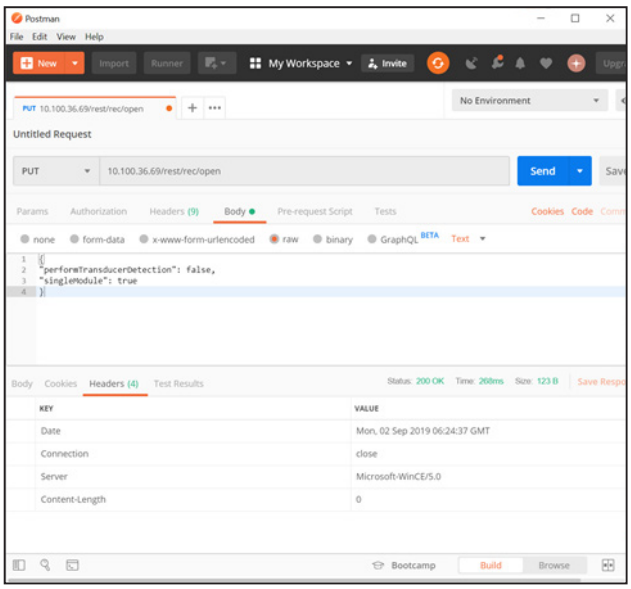

Figure 8. PUT command is sent via Postman.

Please note that when a LAN-XI module is controlled by the Open API, the display still shows correct status. You can even use the small button next to the display to start/stop your ongoing measurement, etc.

STREAMING DATA – USING SAMPLE CODE

The REST/JSON commands that we have dealt with until now, by far cover most of the Programmer's Reference, and probably most of your resulting code. If, however, you look at a typical application, the streaming of data takes 99% or more of the bandwidth on the wire. For the sake of performance, this is binary. On github.com/hbk-world you can find samples on how to parse the binary header data and process the sample data. There are some samples in C#, others in Python. In both cases, we use the http/web libraries that come with the language. A typical flow could be, for example:

- 1. Connect to the LAN-XI module and put it in '**Recorder**' mode (Open API mode).
- 2. Optionally, do a TEDS detect to find the transducers sensitivities, etc.
- 3. Get, or set, the sample-rate and filters.
- 4. Tell the module to stream data to your client (by default data goes to an inserted SD card).
- 5. Create a socket and start receiving blocks of data.
- 6. Parse the header to find metadata and descriptions of channels – including scale factors.
- 7. Buffer samples and display/filter/store as needed.
- 8. Repeat 6 to 7 (might require a threaded implementation) per block of data.

Figure 9 shows the output from a Python sample named 'RealtimePlot.py' at https://github.com/hbk-world/LAN-XI-Open-API-python-examples. This sample is animated and uses threaded code. The top graph is in the time domain and shows a 1 kHz sine wave (simply generated with a phone), while the bottom graph shows an FFT of the signal (using a Hamming window). In this case, a microphone is connected, and the Y-axis shows magnitude in dB re 20µPa (based on TEDS sensitivity).

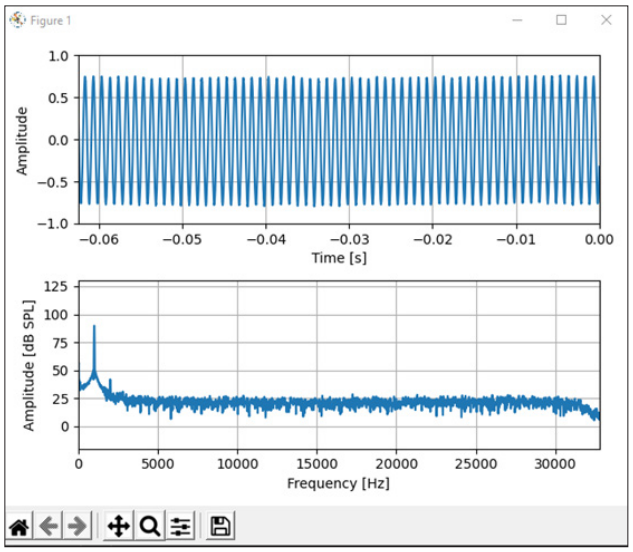

Figure 9. Output from Python sample-program – analyzing a 1 kHz tone from a phone

Figure 9 shows the output from a Python sample named 'RealtimePlot.py' at https://github.com/hbk-world/LAN-XI-Open-API-python-examples. This sample is animated and uses threaded code. The top graph is in the time domain and shows a 1 kHz sine wave (simply generated with a phone), while the bottom graph shows an FFT of the signal (using a Hamming window). In this case, a microphone is connected, and the Y-axis shows magnitude in dB re 20µPa (based on TEDS sensitivity).

If the LAN-XI module has one or more generators, you can also control these. Each generator has two specific modes:

- 1. A classic signal generator with two internal sources. You can, for example, select one source to be a fixed sine, and the other to be a swept sine. The generator will create the two signals, mix them internally, and output the sum-signal on the front connector of the module. This will go on until stopped.
- 2. In the same way that you can stream data from a module, you can stream binary data to the generator in the module, and it will output the analog data on the module's front connector.

Earlier, we saw that the http-based commands are easy to see in Wireshark. If you want to use this tool with the binary data-streams, you will appreciate our 'Wireshark Dissector' found at github.com/hbk-world under ' LAN-XI-Open-API-Tools'.

APPENDIX – GROUND RULES

When using LAN-XI and the Open API there are rules that must be respected:

- LAN-XI supports both Open API and BK Connect/PULSE LabShop – but only one at a time. You should, for example, power-cycle LAN-XI between sessions with the Open API and other clients.
- Multiple Open API clients towards the same LAN-XI module are possible at the same time – but there are constraints. The LAN-XI module can only stream to a single client – or to the SD Card. Other clients may show status and allow users to start or stop measurements.
- http-based commands, by definition, operate on a single module. However it is possible to arm all modules, and then start sample-simultaneous sampling – see code-samples.
- To perform streaming, each module must have appropriate licenses. Please note that there are separate licenses for input- and output-streaming.
- LAN-XI supports neither https nor WebSockets and has not been verified with IPv6. You can, however, select to have one socket per channel, instead of the default, which sends all data on a single socket.
- PTP synchronization requires that modules and frames can 'see' each other via the left connector on frames (rear view). Thus, if you only have two frames, or a frame and a stand-alone module, you can connect your PC to the right connector (rear view) on the frame. Larger setups will require a PTP-aware switch to which the PC can also be connected. Note that some low-quality switches may support the PTP protocol, but still deliver lousy timing. HBK recommends our UL-0265 or high-quality brands like Hirschmann and Cisco.

BN 2516 – 11

Hottinger Brüel & Kjær A/S
Skodsborgvej 307,
2850 Nærum, Denmark
Tel +45 77 41 20 00
www.hbkworld.com
info@hbkworld.com