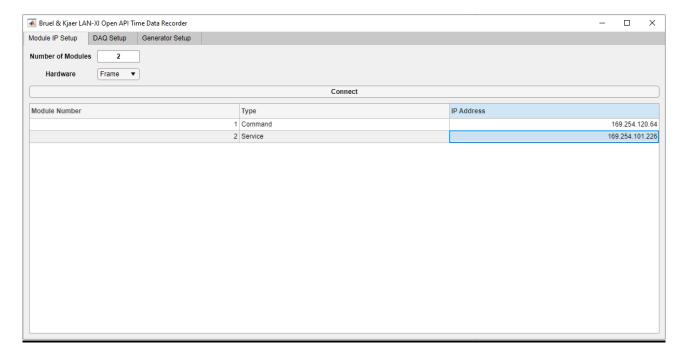
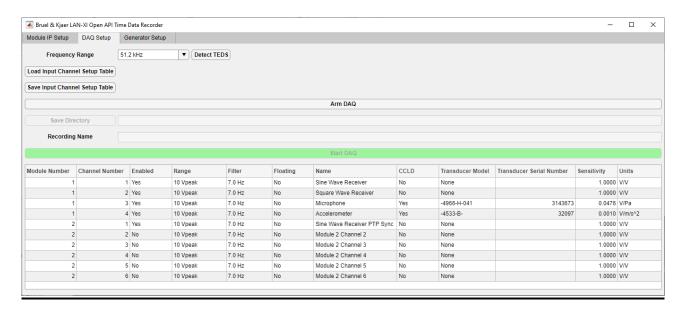
Running B&K Developed Simple Recorder MATLAB Application

- Place the app file (HBKLANXIDAQ.mlapp) and all function files (.m extension) in the same directory
- Create the following sub-directories (names must match these exactly, including case, for code to work)
 - Generator Input Files
 - o Generator Setup Table Files
 - o Input Channel Setup Table Files
 - Measurement Files
- In MATLAB navigate to the directory that contains the app file
- In the command window type 'HBKLANXIDAQ' and then hit enter
- In the Module IP Setup tab:
 - o Type in a numeric value for Number of Modules
 - o Select Switch or Frame for Hardware if Number of Modules > 1
 - Enter IP addresses for each module in the setup, noting that the app forces the command module to be Module Number 1 the same as a B&K Frame would
 - Click Connect, all selections will be grayed out while the connection sequence executes

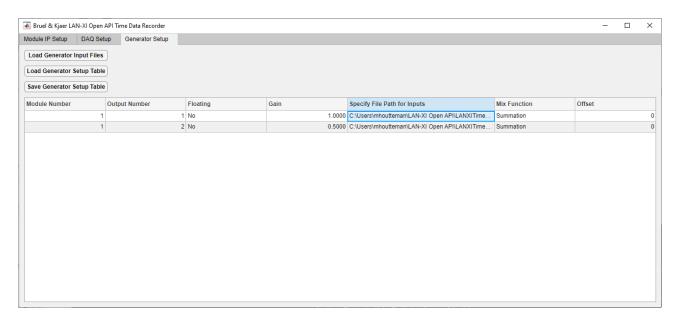


- In the DAQ Setup tab:
 - Select a Frequency Range from the drop-down menu
 - Edit the properties of each channel as needed or load a previously saved input channel setup by clicking 'Load Input Channel Setup Table'
 - o Save input channel setup by clicking 'Save Input Channel Setup Table'
 - Run TEDS detect as required
 - TEDS detect occurs automatically during connection sequence and it is only necessary to rerun it if you change, add, or remove transducers after the connection sequence

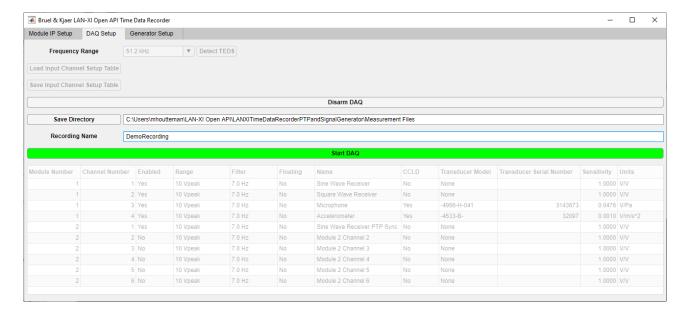
 Note that running TEDS detect will over-write manually entered data for TEDS transducers



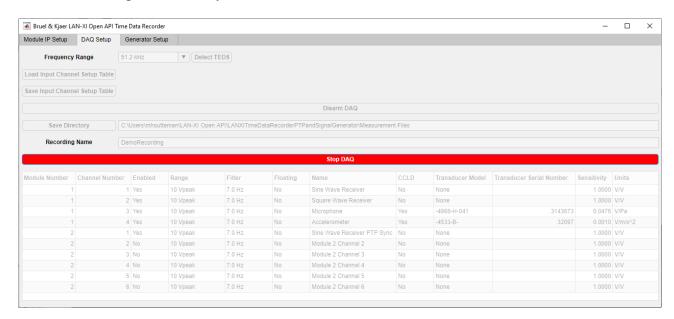
- In the Generator Setup tab:
 - o Edit the properties of each generator channel as needed or load a previously saved generator channel setup by clicking 'Load Generator Setup Table'
 - In the 'Specify File Path for Inputs' field
 - Enter the full file path to the generator waveform .mat file, e.g.
 'C:\Users\mhoutteman\LAN-XI Open
 API\LANXITimeDataRecorderPTPandSignalGenerator\Generator
 Input Files\SineWave100Hz.mat'
 - Alternatively, click on 'Load Generator Input Files' and select a folder that contains generator waveform .mat files to create a dropdown menu of file options in the 'Specify File Path for Inputs' field
 - Save generator channel setup by clicking 'Save Generator Setup Table'



- When setup is complete return to DAQ Setup tab and click 'Arm DAQ'
- Click 'Save Directory' to choose a folder to save measurement files in
- Enter a recording name
 - o No extension, the app will save in .mat format automatically
 - o All MATLAB standard file naming rules apply, e.g. no spaces in filename
- Click 'Start DAQ' when ready



Click Stop DAQ when you wish to end data collection



- At this point you are free to make further recordings by specifying new filenames and/or directories
- To disarm the system click the 'Disarm DAQ' button. Note that you must disarm the system to change the sample rate and/or modify the input or generator channel properties

- To disconnect the system go to the Module IP Setup tab and click 'Disconnect'. Note that disconnecting will reboot all modules in the setup.
- Closing the app at any point will result in all modules being rebooted.
- Allow adequate time for module to display an IP address on the green information panel before attempting to reconnect to module
- Formatting Generator Waveform .mat Files
 - In MATLAB create a cell named 'Inputs1and2' using the following command:
 >>Inputs1and2 = { };
 - O Within the cell create two 1x1 structs:
 - Inputs1and2{1,1} contains the properties for Input 1 of the mixer
 - Inputs1and2{2,1} contains the properties for Input 2 of the mixer
 - The result of mixing the two structs, or the cell Inputs1and2, is the waveform for the output channel
 - Refer to the LAN-XI Open API User Guide for available waveforms and their associated properties and please note that these properties will be different depending on the type of waveform generated
 - Check the formatting of waveforms using the MATLAB function 'jsonencode' to convert the Inputs1and2 cell to a jsonstring: >>jsonencode(Inputs1and2)
 - o Several simple waveforms are included with the code as examples

REST Command Chronology

Connecting Sequence (Idle \rightarrow Connected)

- 1. Read module information using GET command /rest/rec/module/info
- 2. Set module time using PUT command /rest/rec/module/time
 - a. Send to command module first then service modules
- 3. Set synchronization mode using PUT command /rest/rec/syncmode
 - a. Send to command module first then service modules
- 4. Wait for ptpStatus of "Locked" using GET command /rest/rec/onchange
- 5. Open recorder application using PUT command /rest/rec/open
 - a. Send to service modules first then command module
 - b. Wait for moduleState to be "RecorderOpened" using GET command /rest/rec/onchange
- 6. Prepare signal generators using PUT command /rest/rec/generator/prepare
 - a. Send to service modules first then command module
- 7. Get default set-up for all input channels using GET command /rest/rec/channels/input/default
- 8. Get default set-up for all signal generators using GET command /rest/rec/generator/output

TEDS Detection (module must be in 'Connected' state)

- 1. Run TEDS detection using POST command /rest/rec/channels/input/all/transducers/detect
 - a. Send to service modules first then command module
 - b. Wait for transducerDetectionActive to be "false" using GET command /rest/rec/onchange
- 2. Obtain transducer information using GET command -

/rest/rec/channels/input/all/transducers

a. Send to service modules first then command module

Arming Sequence (Connected \rightarrow Armed)

- 1. Configure generator using PUT command /rest/rec/generator/output
 - a. Send to service modules first then command module
- 2. Start generator using PUT command /rest/rec/generator/start
 - a. Send to service modules first then command module
- 3. Synchronize generators using PUT command (this step required only for multi-module systems) /rest/rec/apply
 - a. Send only to command module
- 4. Create new configuration using PUT command rest/rec/create
 - a. Send to service modules first then command module
 - b. Wait for moduleState to be "RecorderConfiguring" using GET command /rest/rec/onchange
 - c. Apply desired input channel configuration to module using PUT command rest/rec/channels/input
 - d. Wait for inputStatus to be "Settled" using GET command (this step required only for multi-module systems) /rest/rec/onchange
- 5. Synchronize modules using PUT command (this step required only for multi-module systems) /rest/rec/synchronize
 - a. Send to service modules first then command module
 - b. Wait for inputStatus to be "Synchronized" using GET command /rest/rec/onchange
- 6. Start internal streaming (multi-module systems only) using PUT command rest/rec/startstreaming
 - a. Send to service modules first then command module
- 7. Get streaming socket using GET command /rest/rec/destination/socket

Measuring (Armed \rightarrow Recording)

- 1. Start streaming using POST command /rest/rec/measurements
 - a. Send to service modules first then command module
 - b. Wait for moduleState to be "RecorderRecording" using GET command /rest/rec/onchange
- 2. Subscribe to binary data stream then run while loop and decode messages until data collection is finished → LAN-XI TCP/IP streaming sequence is as follows
 - a. Message format is header then payload
 - b. Header contains
 - i. Message type
 - ii. Time
 - iii. Length of message content
 - iv. Message data
 - c. Depending on the value received for message type in the header the message is either an interpretation message or a signal data message
 - i. Interpretation messages contain a scale factor to apply to signal data, each channel has its own separate interpretation message
 - ii. Signal data messages contain signal data that needs to be scaled
 - d. Example chronology of messages received for DAQ using three channels
 - i. Interpretation message Channel 1
 - ii. Interpretation message Channel 2
 - iii. Interpretation message Channel 3

- iv. Signal data message Channel 1
- v. Signal data message Channel 2
- vi. Signal data message Channel 3
- vii. Signal data message Channel 1
- viii. Signal data message Channel 2
 - ix. Signal data message Channel 3
 - x. and so on...

Stop Measuring (Recording → Armed)

- 1. Stop streaming using PUT command /rest/rec/measurements/stop
 - a. Wait for moduleState to be "RecorderStreaming" using GET command /rest/rec/onchange

Disarming Sequence (Armed → Connected)

- 1. Stop generator using PUT command rest/rec/generator/stop
- 2. End current recording session using PUT command rest/rec/finish
 - **a.** Wait for moduleState to be "RecorderOpened" using GET command /rest/rec/onchange

Disconnecting Sequence (Connected \rightarrow Idle)

- 1. Close recorder application using PUT command /rest/rec/close
 - a. Send to service modules first then command module
 - b. Wait for moduleState to be "Idle" using GET command /rest/rec/onchange
- 2. Set synchronization mode back to defaults using PUT command /rest/rec/syncmode
 - a. Send to service modules first then command module

Rebooting

- 1. Reboot module using PUT command rest/rec/reboot
 - a. Send to service modules first then command module