Calling KULT User Libraries

The 4200A-CVIV and the PMU measuring capability do not have KXCI commands.

To control them remotely you can call KULT user libraries in KXCI.

KXCI includes a list of commands to call user libraries that were built in the KULT application on the 4200A-SCS.

These commands include:

- **UL** – Switches KXCI to the user library mode
- **EX** – Executes the specified user module
- **GN** – Get parameters as specified by name
- **GP** – Get parameters specified by a number
- **GD** – Get description of the specified function

The CVIV and PMU measurement capability do not have KXCI commands. But you can control them by calling KULT user libraries in KXCI. Most user libraries are able to be called in KXCI.

Five commands are provided to enable you to call user libraries. The UL command switches KXCI to the user library mode. The EX command is used to specify the user module. The GN and GP commands get the input and output parameters as specified. And, finally, the GD command gets the description of a specific function.

Calling a User Library and User Module

UL: Switches KXCI Operation to call user libraries
- Must be sent before any of the other library commands

EX: Executes a user module with the specified parameters
- Example: `EX libName modName(param1, param2, param3)`
- Parameter syntax
  - Parameters are separated by commas
  - Values in an array are separated by semicolons
    i.e. `param1, 0;0;0;0, param3` where parameter 2 is an array of 4 0's
  - String parameters should not have spaces before the string
    i.e. `string1,string2` not `string1, string2`
  - Input and output arrays should be less than 16,000 values
  - Output variables should be left blank in the parameter list
    i.e. `param1, , param 3` where parameter 2 is an output value

66

The two most important commands to use when calling a user library are the UL and EX commands.

The UL command switches KXCI operation to the usrlib mode.

The EX command executes a user module using specified parameters as shown in this slide.

After the EX, enter the User Library name, user module name, then input all the parameters as listed in the user module separated by commas. Values in an array are separated by semicolons. String parameters should be separated by commas only, without leading spaces to prevent parsing errors. Output parameters are just left as blank spaces separated by commas.

Let's do an example.

Example Program

Controlling the CVIV through KXCI

KCXI Commands

- This code connects the CV HI and CV LO to channels 1 and 2, and call the test CV MEAS:

```
UL
EX cvivulib cviv_configure (CVIV1, 1, 2, 3, 0, 0, CVHI, CVLO, NC, NC, CV
Meas, )
```

Module Parameters:

```
InstId,   char *,  Input,    "CVIV1",  ,
                   TwoWireMode,       int,      Input,    1,        0,        1
                   Ch1_Mode, int,     Input,    1,        0,        12
                   Ch2_Mode, int,     Input,    1,        0,        12
                   Ch3_Mode, int,     Input,    1,        0,        12
                   Ch4_Mode, int,     Input,    1,        0,        12
                   Ch1_TermName,      char *,   Input,    "One",    ,
                   Ch2_TermName,      char *,   Input,    "Two",    ,
                   Ch3 _ TermName,    char *,   Input,    "Three",  ,
                   Ch4_TermName,      char *,   Input,    "Four",   ,
                   TestName, char *,  Input,    "CVIV Test",      ,
                   ConstantsName,     char *,   Output,   ,         ,
```

The commands shown can be sent regardless of communication type. The user module used is the cviv_configure module, whose parameters are shown at the bottom. Note that the last parameter, ConstantsName, is an output parameter. Therefore, we leave a blank space at the end of the call to account for that parameter.

Return Values

```
2020/08/13 - 16:21:01  INPUT: EX cvivulib cviv_configure (CVIV, 1, 2, 3, 0, 0, CVHI, CVLO,..
2020/08/13 - 16:21:02  Return Value = -1
```

```
2020/08/13 - 16:36:02  INPUT: EX cvivulib cviv_configure (CVIV1, 1, 2, 3, 0, 0, CVHI, CVLO...
2020/08/13 - 16:36:04  Return Value = 0
```

```
Return values
-------------
Value   | Description
------  | -----------
0       | Pass
-1      | Invalid CVIV instrument Id
-2      | Display configuration error
-78     | Receive Timeout Waiting for CVIV Response (LPT)
-88     | Bad Configuration of Data Sent to CVIV (LPT)
-122    | Invalid parameter (LPT)
-150    | CVIV Device Not Found (LPT)
-167    | Invalid CVIV Connection Configuration (LPT)
-169    | CVIV Unit Name not configured (LPT)
```


Once the user module is executed, the Return Value status of the user module will be returned to the KXCI screen just like they are returned to Clarius in the first column of the sheet. These return values indicate whether there was an error during the code execution.

Each Keithley provided user module has a list of Return Values that can be found in the source code or the Help pane in Clarius.

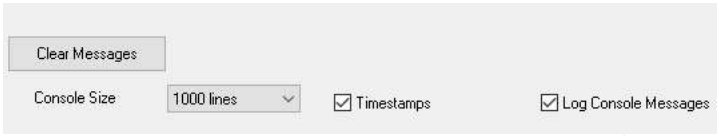
In this CVIV example, the instrument ID was inputted incorrectly. The correct Id is CVIV1 and not just CVIV. This caused a Return Value of -1. When the ID is corrected and the module is re-run, a value of 0 is returned, indicating that the code ran successfully.

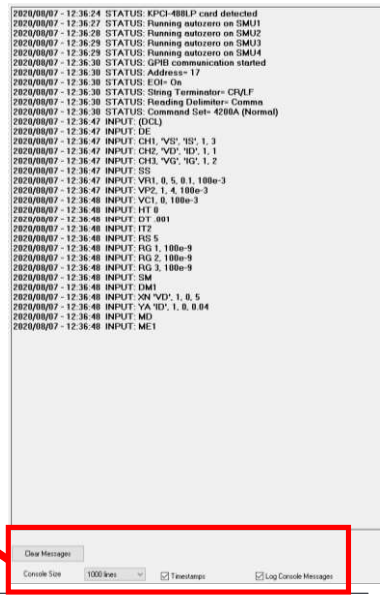
KXCI Status

- KXCI Log File
- Error Messages

KXCI Log File

- Clear messages in the KXCI window.
- Add timestamps to each line
- Log console messages:
  Messages are saved to file named KXCILOGfile.txt, located at C:\s4200\sys\KXCI\





The KXCI console logs message during startup, execution and data return so that you can monitor the status of your test. At the bottom of the KXCI console are a few settings. One button is used to clear the messages listed in the console.

You can also choose to add a timestamp to each line. This screen capture shows the time stamps turned on.

Finally, you can also save the logged messages. These are saved on the 4200A-SCS at the location listed on this slide.

Error Messages

Error Number	Error Message	Error Number	Error Message
-999	IEEE32.DLL GPIB driver is not loaded	-991	Illegal setup error
-998	Unable to initialize shared memory	-990	Trigger Master card not found
-997	Could not establish communication with console	-989	Command not valid on this page
-996	GPIB address not sent as argv{1}	-988	Instrument not mapped
-995	GPIB address not in 0<=addr<=31	-987	Skipping instrument
-994	Could not find configuration file	-986	Unsupported command received
-993	GPIB argument error	-985	Unsupported file format error
-992	GPIB command error	-984	Could not open specified file

If an error is generated during the test, an error message can be returned to the KXCI message console. This table lists some of the error messages that can be generated.

The most common error is the -993 GPIB argument error because it’s generated when a syntax error occurs.

Error Messages in the KXCI Console

```
2020/08/10 - 12:34:04  INPUT: :CVU:RESET
2020/08/10 - 12:34:04  INPUT: :CVU:MODE 1
2020/08/10 - 12:34:04  INPUT: :CVU:MODEL 2
2020/08/10 - 12:34:04  INPUT: :CVU:SPEED 2
2020/08/10 - 12:34:04  INPUT: :CVU:ACZ:RANGE 0
2020/08/10 - 12:34:04  INPUT: :CVU:FREQ 1E6
2020/08/10 - 12:34:04  INPUT: :CVU:SWEEP:DCV: 5, -5, -0.2
2020/08/10 - 12:34:04  ERROR: GPIB argument error. (-993)
```

No ":"
This is correct

```
2020/08/10 - 12:37:53  INPUT: :CVU:SWEEP:DCV 5, -5, -0.2
```


Here is an example of a gpib argument error -993. Error messages are listed in the KXCI window directly after the command that caused the error.

In this example there is an added colon after DCV that is causing the error. Once the colon is removed as shown at the bottom of the slide, the error is gone.