Arkeo Multichannel API

v0.3

Contents

Introduction

This API facilitates remote control of the Arkeo Multichannel using TCP on port 6340, enabling users to perform a variety of commands manage measurements.

A command must sent as a JSON string in the form
`{'command': command, 'parameter': parameter}`
Each command must be preceded by a 4-byte integer containing the total length of the message.
After each command, a reply is sent back. Its format depends on the command (see list below)

Summary of available commands

Command	**Description**
SetActiveChannel	Set the channel upon which all succeeding actions are performed
GetActiveChannel	Get the active channel
SetChannelSettings	Modify the settings
GetChannelSettings	Retrieve the settings
StartChannel	Start measurement
StopChannel	Stop the measurement process
ForceJV	Force a JV measurement
GetChannelState	Retrieve the current state

List of commands

Command: SetActiveChannel

Description: Set the channel upon which all succeeding actions are performed.
Parameters:

- channel_id (integer): The ID of the channel.

Response:

- channel_id (integer): The ID of the channel.

Command: GetActiveChannel

Description: Get the active channel.
Parameters:

Response:

- channel_id (integer): The ID of the channel.

Command: SetChannelSettings

Description: Set the JSON configuration string for active channel.
Parameters:

- settings (JSON string): The configuration settings in JSON format (see chapter: The JSON settings string).

Response:

- status (string): OK

Command: GetChannelSettings

Description: Retrieve the JSON configuration string of active channel.
Parameters:

Response:

- settings (JSON string): The configuration settings in JSON format (see chapter: The JSON settings string).

Command: StartChannel

Description: Start the measurement process for active channel.
Parameters:

Response:

- status (string): OK

Command: StopChannel

Description: Stop the measurement process for active channel.
Parameters:

Response:
- status (string): OK

Command: ForceJV

Description: Force a JV measurement on active channel if channel is in tracking mode.
Parameters:
- channel_id (integer): The ID of the active channel.

Response:
- status (string): OK

Command: GetChannelState

Description: Retrieve the current state of a specific channel (e.g., running, stopped, JV, tracking).
Parameters:

Response:
- state (string): JSON of the state of the active channel (see chapter: The JSON state string)

Command: GetLatestJV

Description: Retrieve the voltage and current values of the latest JV.
Parameters:

Response:
- JV Data (string):
  v_fw1|j_fw 1|...|v_fw n|j_fw n|| v_rv1|j_rv 1|...|v_rv n|j_rv n

Note:

Forward and Reverse scans are separated by || (double pipe character)
Voltage and Current arrays are interleaved and separated by | (single pipe character)
Voltage unit: V

Current unit A/cm^2

Command: GetIV

Description: Retrieve the live voltage and current values of all channels.
Parameters:

Response:

- Voltage and Current values (string):
  v1|j1|...|vn|jn

Note:

Voltage and Current arrays are interleaved and separated by | (single pipe character)
Voltage unit: V
Current unit A/cm^2

Command: GetSensors

Description: Retrieve the live voltags of all sensors.
Parameters:

Response:

- Sensor values (string):
  v1|...|vn|

Note:

Sensor voltages are separated by | (single pipe character)
Voltage unit: V

The JSON settings string

The settings of a channel are represented in a JSON string. When using the **Command: SetChannelSettings**, this is value that the software expects. It is recommended to use the **Command: GetChannelSettings** to read the actual settings and modify those settings.

Example

Below is an example of a channel settings string. A detailed description can be found on the next page.

```
{
  "Enable":true,
  "User":"User",
  "Device":"Sample",
  "JV":{
    "Vmin (V)":-0.1,
    "Vmax (V)":2,
    "VocDetect":true,
    "Overvoltage":0,
    "Step (mV)":20,
    "ScanRate (mV/s)":100,
    "ScanOrder":"FW then RV",
    "VoltageLimit":"10 V",
    "CurrentLimit":0,
    "InvertedStructure":false
  },
  "Tracking":{
    "TrackEnable":true,
    "Algorithm":"MPPT",
    "dV (V)":0.01,
    "jvInterval":0.1,
    "jvIntervalUnit":1,
    "TestDuration":100,
    "DurationUnit":1,
    "ConstantOutput":0.2,
    "SaveDecimation":10
  },
  "Cell":{
    "Type":"Cell",
    "Area (cm²)":1,
    "NrCells":1,
    "NrW cells":1,
    "W cell area":1
  },
  "Note":""
}
```

Description

PARAMETER	DESCRIPTION	EXAMPLE	UNIT	DATA TYPE
Enable	Enables or disables the channel	true		Boolean
User	Name of the user	User		String
Device	Device Name	Sample		String
JV				Object
- Vmin	Minimum voltage	-0.1	V	Float
- Vmax	Maximum voltage	2	V	Float
- VocDetect	Enables the detection of open-circuit voltage	true		Boolean
- Step	Voltage increment per step	20	mV	Integer
- ScanRate	Rate at which the voltage is scanned	100	mV/s	Integer
- ScanOrder	Order of scanning: *0: FW then RV* *1: RV then FW* *2: Forward Only* *3: Reverse Only*	0		Integer
- VoltageLimit	Maximum allowable voltage during testing	10	V	Float
Tracking				Object
- TrackEnable	Enables tracking	true		Boolean
- Algorithm	Specifies the algorithm used for tracking	MPPT		String
- dV	Voltage differential for the tracking algorithm	0.01	V	Float
- jvInterval	Time interval between JV measurements	10	*	Float
- jvIntervalUnit	Unit for JV interval 0: minutes 1: hours	0		Integer
- TestDuration	Total duration of the tracking test	100	*	Float
- DurationUnit	Unit for Test Duration 0: minutes 1: hours	1		Integer
- ConstantOutput	Setting to maintain a constant output	0.2	*	Float
- SaveDecimation	Determines how often data is saved	10		Integer
Cell				Object
- Area	Area of the cell	1	cm^2	Float
- NrCells	Number of cells	1		Integer
- NrW cells	Number of W cells	1		Integer
- W cell area	The area of each W cell	1	cm^2	Float

The JSON state string

The state of a channel is represented in a JSON string. This string is read-only and is obtained using the **Command: GetChannelState**.

Example

Below is an example of a state string. A detailed description can be found below.

```
{"Enable":false,
  "Channel":"1A",
  "User":"User",
  "Measurement":"JV",
  "Direction":"Forward",
  "State":"Running",
}
```

Description

PARAMETER	DESCRIPTION	EXAMPLE	DATA TYPE
Enable	Enables or disables the channel	true	Boolean
Channel	ID of the channel	1A	String
User	Name of the user	User	String
Measurement	State of the Measurement	JV	String
Direction	Current direction of the JV	Forward	String
State	Current state of the channel – Idle – Ready to start – Running – Paused – Stopped – Error	Running	String

Examples

Python

```python
import socket
import json

def exampleCommand(command, data=''):
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as sock:
        sock.connect(('localhost', 6340))

        message = json.dumps({'command': command, 'data': data})

        # Prepare the length of the string
        length = len(message)
        length_bytes = length.to_bytes(4, byteorder='big')  # 4 bytes to
represent the length

        sock.sendall(length_bytes) # Send the length of the string
        sock.sendall(message.encode('utf-8')) # Send the string

        # Receive response
        response = sock.recv(1024)
        print('Received:', response.decode('utf-8'))

exampleCommand('GetChannelSettings')
```