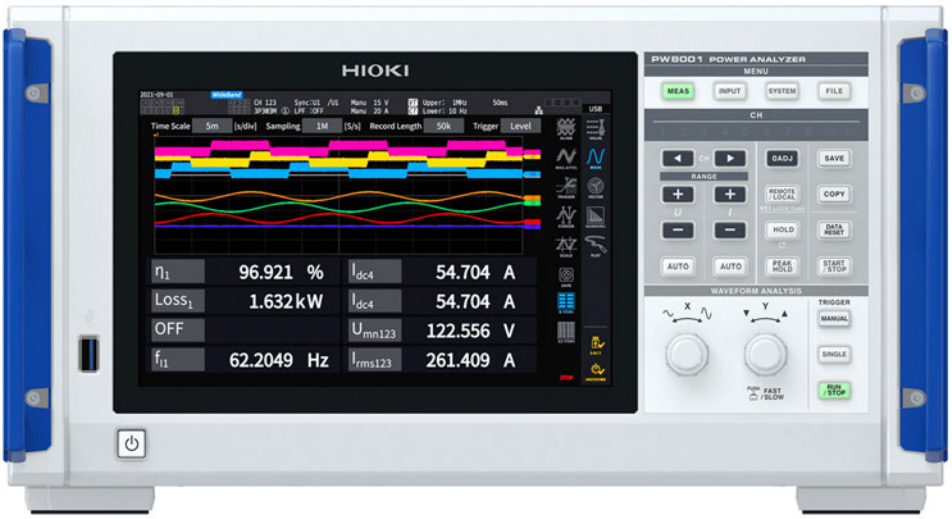
PW8001

HIOKI

PW8001-01
PW8001-02
PW8001-03
PW8001-04
PW8001-05
PW8001-06
PW8001-11
PW8001-12
PW8001-13
PW8001-14
PW8001-15
PW8001-16

MATLAB Toolkit User's Manual

POWER ANALYZER



EN

Dec. 2022 Edition 1
PW8001A970-00 22-12H

- ✓ This instruction manual deals only with the parts related to commands.
- ✓ Before using the PW8001, be sure to read the PW8001 Instruction Manual.
- ✓ For information on the communication settings of the PW8001, please refer to section ″9 Connection with PC″ in the PW8001 Instruction Manual.
- ✓ While every effort has been made to ensure the accuracy of the contents of this instruction manual, if you have any questions or notice any errors, please contact our call center at the head office or the nearest sales office.

Contents

1 Overview

This software consists of a MATLAB script (tcpipPW8001.p) to control our power analyzer PW8001 connected via Ethernet with MATLAB.

It is recommended to use this toolkit with MATLAB R2022a or later. In addition, this toolkit cannot be used to control PW8001 with MATLAB using a communication interface other than Ethernet (GPIB, RS-232C).

2 Ready for use

First, make sure you have this manual file (Power Analyzer PW8001 Matlab Toolkit User's Manual.pdf) and the script file (tcpipPW8001.p) in MatlabToolkitForPW8001.zip and place them in an appropriate location.

And, add the location to the MATLAB search path.

*MATLAB is a registered trademark of The MathWorks, Inc.

3 Control PW8001 on MATLAB via Ethernet connection

By using the tcpipPW8001 class provided by this toolkit, communication commands (see next page for details) are sent from MATLAB to the Ethernet-connected PW8001 and its response is received.

Examples of execution are shown in Figures 1 and 2. In Figure 1, analog waveform data is acquired. The procedure is described below.

1. Sets the IP address of PW8001 and the communication timeout period in MATLAB and generates an object.
2. Establishes an Ethernet connection between the generated object and the PW8001.
3. Sends a communication command from MATLAB to PW8001 and receives a response from PW8001 to the communication command.
4. Selects one analog waveform to be acquired from PW8001 and captures the maximum and minimum values of the data.
5. Displays the acquired data with the plot function.
6. Disconnects the Ethernet connection.

Executes steps 1. through 6. in the command window of MATLAB, resulting in the following process.

```
>> [obj, flag] = tcpipPW8001(″192.168.10.11″, 15);
% Generates a TCP/IP object for Ethernet connection with IP address of PW8001 and timeout period set to 15 seconds
>> obj.open;  % Ethernet connection to PW8001
>> obj.send(″*IDN?″);  %Sends the *IDN? command
>> obj.receive;  %Receives command response
>> [samplingSpeed, storageLength, storageMode, waveDataMax,...
    waveDataMin, flag] = obj.DownloadAnalogWaveData(″U1″);  % Captures voltage waveform data of U1
>> plot(waveDataMax);  % Plots the data sequence of the maximum value of the voltage waveform of U1
>> obj.close;  %Disconnects the Ethernet connection
```

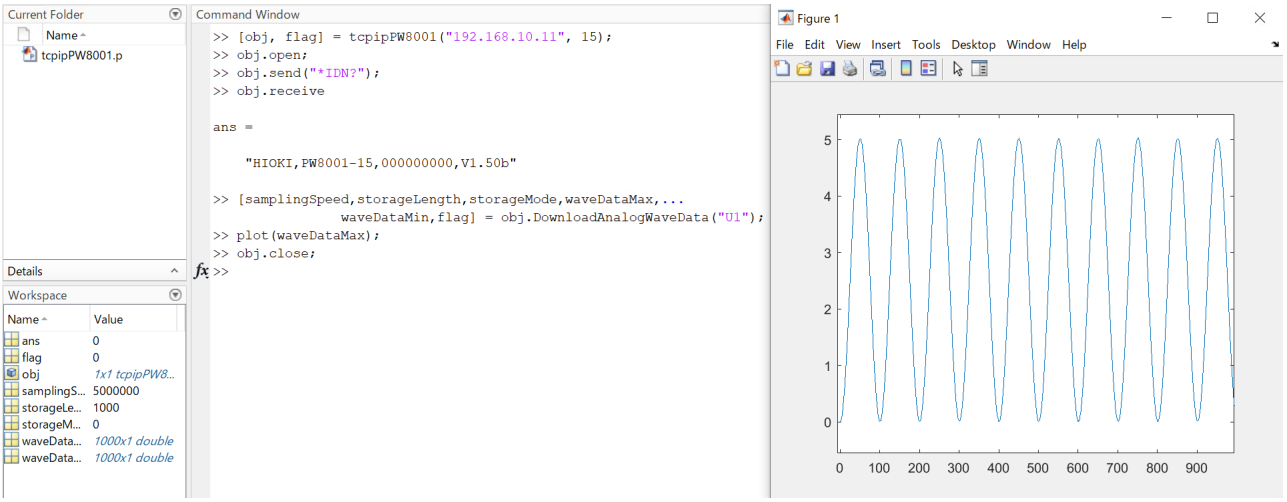


Figure1：Execution example of acquiring analog waveform data (controlling PW8001 with MATLAB via Ethernet connection)

Next, in Figure 2, pulse waveform data is acquired. Pulse waveform acquisition is available only when the PW8001 is equipped with the motor option. The procedure is described below.

1. Sets the IP address and communication timeout period of PW8001 in MATLAB and generates the object.
2. Establishes an Ethernet connection between the generated object and PW8001.
3. Sends a communication command from MATLAB to PW8001 and receives a response from PW8001 to the communication command.
4. Captures information on the motor channels set for pulse input from PW8001, as well as the maximum and minimum data values for all those channels.
5. Selects one channel from the data of the motor channels set for pulse input, selects the maximum or minimum value of the pulse waveform data, and acquires it.
6. Displays the acquired data with the plot function.
7. Disconnects the Ethernet connection.

Executes steps 1. through 7. in the command window of MATLAB, resulting in the following process.

```
>> [obj, flag] = tcpipPW8001(″192.168.10.11″, 15);
% Generates a TCP/IP object for Ethernet connection with IP address of PW8001 and timeout period set to 15 seconds
>> obj.open;  % Ethernet connection to PW8001
>> obj.send(″*IDN?″);      %Sends the *IDN? command
>> obj.receive;    %Receives command response
>> [samplingSpeed, storageLength, storageMode, logicCH,...
                    waveDataMax, waveDataMin, flag] = obj.DownloadLogicWaveData;
% Obtains data including all waveforms set for pulse input.
>> [chWaveData, flag] = obj.ExtractLogicChWaveData(″CHA″,waveDataMax);
% Captures the maximum value of CHA pulse waveform data
>> plot(chWaveData);    % Plots a data sequence of the maximum values of the voltage waveforms of CHA
>> obj.close;    % Disconnects the Ethernet connection
```

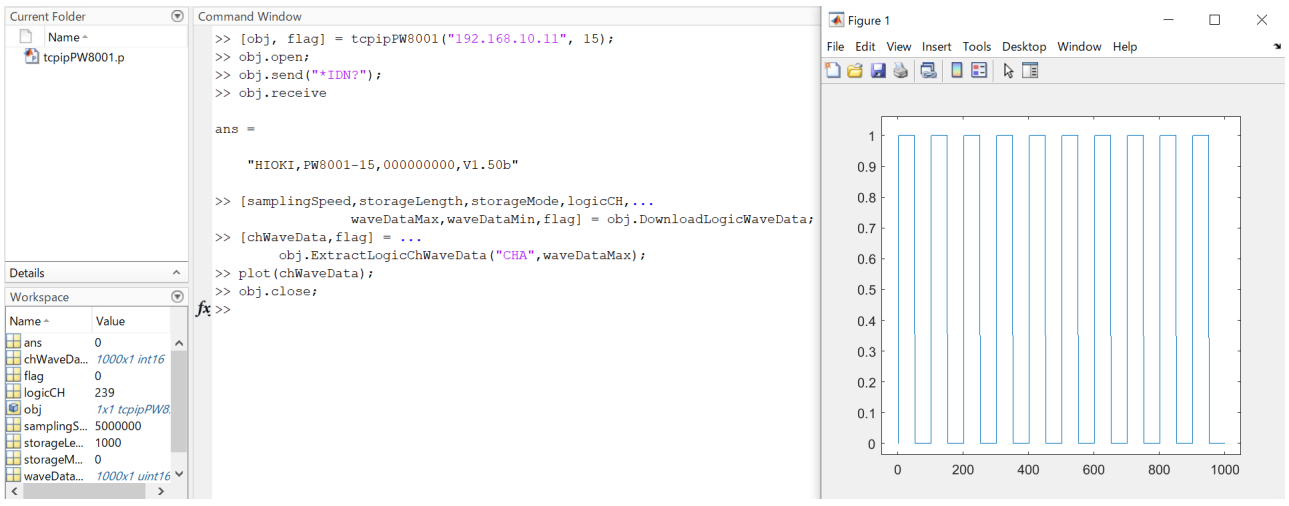


Figure2：Execution example of acquiring pulse waveform data (controlling PW8001 with MATLAB via Ethernet connection)

4 tcpipPW8001Class

This class is used to control PW8001 with MATLAB via Ethernet connection. Through this class, communication commands can be sent to and received from the PW8001 and waveform data can be acquired.

Member Functions

[obj, flag] = tcpipPW8001(ipAddr, timeOut);	
Argument	ipAddr: String of IP address of PW8001 timeOut: Timeout time (seconds) value
Return Value	obj: Generated tcpipPW8001 object flag: "0" if the object could be generated, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Constructor for tcpipPW8001 class object; creates a tcpipPW8001 object to connect to PW8001 and returns the flag.
Example	[obj, flag] = tcpipPW8001("192.168.10.11", 15); % Generates a TCP/IP object for Ethernet connection with IP address of PW8001 and timeout period set to 15 seconds.

[ipAddress, flag] = obj.ipAddr;	
Argument	None
Return Value	ipAddress: String of IP address of PW8001 flag: "0" if IP address could be obtained, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Returns the IP address string for PW8001 and the flag.
Example	obj.ipAddr;

[aTime, flag] = obj.timeout	
Argument	None
Return Value	aTime: Timeout time for communication with PW8001 (sec) flag: "0" if timeout time could be obtained, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Returns the timeout time (in seconds) for communication with PW8001 and the flag.
Example	obj.timeout;

flag = obj.setTimeout(aTimeout)	
Argument	aTimeout: Variable that sets the timeout period for communication with PW8001
Return Value	flag: "0" if timeout time could be set, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Sets the timeout period (in seconds) for communication with PW8001 and returns the flag.
Example	obj.setTimeout(15); % Sets the timeout period for communication with PW8001 to 15 seconds

flag = obj.open	
Argument	None
Return Value	flag:"0" if the connection between the generated object and PW8001 can be established, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Establishes a connection between the generated object and PW8001 and returns the flag.
Example	obj.open;

flag = obj.close	
Argument	None
Return Value	flag:"0" if the connection between the generated object and the PW8001 can be disconnected, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Disconnects the generated object from PW8001 and returns the flag.
Example	obj.close;

flag = obj.send(command)	
Argument	command:Communication command string to be sent to PW8001
Return Value	flag:"0" if the command could be sent to PW8001, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Sends a communication command string to PW8001 and returns the flag.
Example	obj.send("*IDN?"); % Sends the communication command "*IDN?" to PW8001

[str, flag] = obj.receive	
Argument	None
Return Value	str:Responses string of PW8001 to a communication command sent from MATLAB flag:"0" if a response is received from PW8001, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Receives the PW8001 response string to the communication command sent and returns the flag.
Example	obj.receive;

[samplingSpeed, storageLength, storageMode, waveDataMax,...
waveDataMin, flag] = obj.DownloadAnalogWaveData(chName)

Argument	chName : String of the following target waveform names to be retrieved from PW8001 (U1, U2, U3, U4, U5, U6, U7, U8, I1, I2, I3, I4, I5, I6, I7, I8, CHA, CHC, CHE, or CHG) (However, CHA, CHC, CHE, and CHG are limited to analog waveforms)
Return Value	samplingSpeed : Sampling rate of waveform data storageLength : Number of points of waveform data storageMode : Storage mode of waveform data (Returns 0 for peak compression and 1 for simple thinning as a number.) waveDataMax : Array to store the maximum value of the specified analog waveform data waveDataMin : Array to store the minimum value of the specified analog waveform data flag : "0" if analog waveform data could be obtained from PW8001, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Obtains the maximum and minimum values of the specified analog waveform data from the PW8001 and returns the flag.
Example	[samplingSpeed, storageLength, storageMode, waveDataMax, waveDataMin, flag] = obj.DownloadAnalogWaveData("U1"); % Captures the maximum and minimum voltage waveform data of U1

[samplingSpeed, storageLength, storageMode, logicCH,...
waveDataMax, waveDataMin, flag] = obj,DownloadLogicWaveData

Argument	None
Return Value	samplingSpeed : Sampling rate of waveform data storageLength : Number of points of waveform data storageMode : Storage mode of waveform data (Returns 0 for peak compression and 1 for simple thinning as a number.) logicCH : Motor channel set to pulse input ※Bit correspondence between logicCH and motor channel

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CHH	CHG	CHF	CHE	CHD	CHC	CHB	CHA

waveDataMax : Array to store the maximum value of the data including all waveforms of the motor channel set for pulse input
waveDataMin : Array to store the minimum value of the data including all waveforms of the motor channel set for pulse input
※Bit correspondence between waveDataMax, waveDataMin and motor channel

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
CHA	CHB	CHC	CHD	CHE	CHF	CHG	CHH
bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
–	–	–	–	–	–	–	–

flag : "0" if data including all waveforms of the motor channel set for pulse input can be acquired from PW8001, "1" or a predefined MATLAB error message is displayed in case of abnormal termination.

Function	Obtains data including all waveforms set for pulse input from PW8001 and returns the flag.
Example	[samplingSpeed, storageLength, storageMode, logicCH, waveDataMax, waveDataMin, flag] = obj.DownloadLogicWaveData;

[chWaveData, flag] = obj.ExtractLogicChWaveData(chName, waveData)	
Argument	chName：String of motor channel name set for pulse input (CHA, CHB, CHC, CHD, CHE, CHF. CHG. or CHH) waveData：Either waveDataMax or waveDataMin obtained with the DownloadLogicWaveData function
Return Value	chwaveData：Array to store the maximum or minimum pulse waveform data for the specified motor channel flag："0" if pulse waveform data of the specified motor channel could be acquired from PW8001, "1" or a predefined MATLAB error message in case of abnormal termination
Function	Obtains the maximum or minimum pulse waveform data of the specified motor channel from PW8001 and returns the flag.
Example	[chWaveData, flag] = ExtractLogicChWaveData(obj, "CHA", waveDataMax); % Captures the maximum value of CHA pulse waveform data

HIOKI

www.hioki.com/

All regional contact information

HEADQUARTERS
81 Koizumi
Ueda, Nagano 386-1192 Japan

HIOKI EUROPE GmbH
Helfmann-Park 2
65760 Eschborn, Germany
hioki@hioki.eu

2111 EN

Edited and published by HIOKI E.E. CORPORATION Printed in Japan