POWER-OVER-ETHERNET
SYSTEM ATTENUATOR
MODEL 624

USER GUIDE

Ver 3.0

September 2025

GENERAL INFORMATION

WARRANTY

Flann Microwave Ltd warrants each product it manufactures to be free from defects in material and workmanship. Our obligation under this warranty is limited to servicing or adjusting any products returned to our address for that purpose and to make good at our facility any part or parts thereof (including power supplies, transistors, integrated circuits, batteries, diodes and displays) within one year after making delivery to the original purchaser and which after our examination we agree to have been defective. Such returns must have prior authorization from Flann Microwave Ltd and must be returned as per our detailed instructions with transportation charges prepaid. Warranty returns or repairs must first be authorised by Flann Microwave Ltd. Flann Microwave Ltd does not authorise any third party to assume for them any other liability in connection with the original sale than the foregoing. ***Unauthorized tampering with sealed screws will invalidate the warranty and may result in damage to the product.***

DESIGN CHANGES

Flann Microwave Ltd reserves the right to make changes in the design of its products without notice and without incurring any obligation to make the same alterations on products previously purchased.

SPECIFICATION CHANGES

Flann Microwave Ltd reserves the right to change any specification noted herein without prior notice.

REPAIRS

When wishing to return instruments for repairs, or for any other reason, please contact Flann Microwave Ltd for shipping instructions. To expedite repair service, it is important to provide type number, serial number and a detailed description of the reason, including all fault symptoms, for the return of the instrument.

Flann Microwave Ltd
Dunmere Road
Bodmin
Cornwall
PL31 2QL
United Kingdom

Tel: +44 (0) 1208 77777
Email: sales@flann.com
http: //www.flann.com

ENVIRONMENTAL CONSIDERATIONS

The attenuator is designed to operate in 'non-hazardous' areas. The environment in which the product is to be used is commercial, light industrial, either indoors or in a protected outdoor environment.

The operating environment must conform to the conditions shown in Table 1. Operation outside these ranges cannot be guaranteed and may pose dangerous implications to the operator or cause mechanical or electrical failure to the equipment. ***The device will generate heat during operation; it is important to always maintain adequate ventilation or cooling.***

Environmental Condition	Ranges
Temperature operating	+5 to +35 °C
Temperature non-operating	0 to +40 °C
Humidity operating (max)	90 % without condensation
Humidity non-operating (max)	95 % without condensation

Table 1: Environmental Considerations

GETTING STARTED

INITIAL SET UP AND CONNECTION TO A NETWORK (Pre-August 2025)

The 624 is a Power-over-Ethernet (PoE) device, when connecting the device into a network ensure that the switch or router is capable of providing PoE to IEEE 802.3af.

The Device Name and the MAC address are shown on the instrument label. When supplied from the factory, the device is configured to connect automatically to a network using the DHCP protocol and the IP address will be assigned by the network at the time of connection.

We advise that someone with Administrator access to the network ensures that the instrument is configured before the device is deployed within a network. This person should plug the device into a network and once it has finished negotiating with the network that person will be able to use the MAC address to find the IP address that has been assigned to the device.

OPTIONAL SETUP TO ESTABLISH A STATIC IP ADDRESS (Post-August 2025)

Once the dynamic IP address of the 624 is established and you are able to communicate with the instrument, there is an option to establish static IPv4 addressing parameters. This means that the device can be deployed onto a non-DHCP network and will not seek to negotiate continuously with the network.

The commands used when configuring the IPv4 parameters for a static IP address, Netmask, Gateway and DNS for the device are as shown below. A terminal programme such as (PUTTY) or a Programming Language (like Python) can be used to deliver these commands.

ZI*{IP ADDRESS}*
Sets the Pending IP address to ***{IP ADDRESS}***
Value Example: 123.123.123.123, all sections must be present and between 0 and 255

ZM*{NETMASK ADDRESS}*
Sets the Pending Netmask address to ***{NETMASK ADDRESS}***
Value Example: 123.123.123.123, all sections must be present and between 0 and 255

ZG*{GATEWAY ADDRESS}*
Sets the Pending Gateway address to ***{GATEWAY ADDRESS}***
Value Example: 123.123.123.123, all sections must be present and between 0 and 255

ZD*{DNS ADDRESS}*
Sets the Pending DNS address to ***{DNS ADDRESS}***
Value Example: 123.123.123.123, all sections must be present and between 0 and 255

ZS
This command will return the Current Set of parameters:

Current	Saved Static IP Address
Current	Saved DNS Address
Current	Saved Gateway Address
Current	Saved Netmask Address
Pending	IP Address
Pending	DNS Address
Pending	Gateway Address
Pending	Netmask Address
Value	No Value required

ZC
Transfers all Pending addresses into the Saved addresses. Commits to the Static Addressing.
Value No Value required

Note: **A value needs to be entered using ZI, ZM, ZG and ZD before this command can function, A value is required in each of the 'Pending Addresses' before a ZC command can be executed.**

ZB

Performs a software reset and brings the 624 up using the most recently saved addresses.

Value No Value required

Note: this is the last step required to make the Address STATIC. Alternatively, you could power off the Instrument and restart it.

ZX

Clears the Saved addresses and changes the device from static IP to dynamic addressing

Value No Value required

Some examples of interactions when making the IP address static might look like:

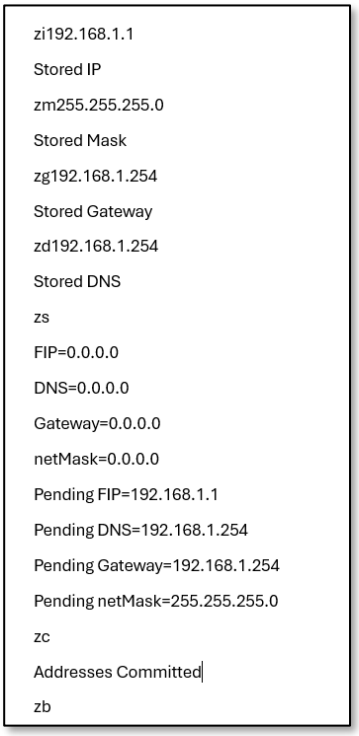

Image 1: Sample Interaction 1

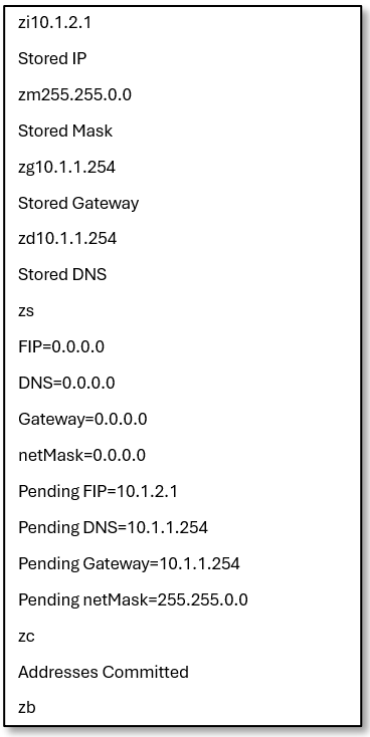

Image 2: Sample Interaction 2

CONTROLLING AND USING THE INSTRUMENT

Once the IPv4 addresses for the device are configured it is possible to communicate and control it through a terminal program (for example PuTTY) or a script that is written using a programming language (for example Python).
The examples below show how to establish communication with the attenuator:

RAW – PuTTY Example

To control the instrument using a RAW connection using the PuTTY Terminal software:

Enter the IP address, e.g. 10.1.2.47 and use port 82:

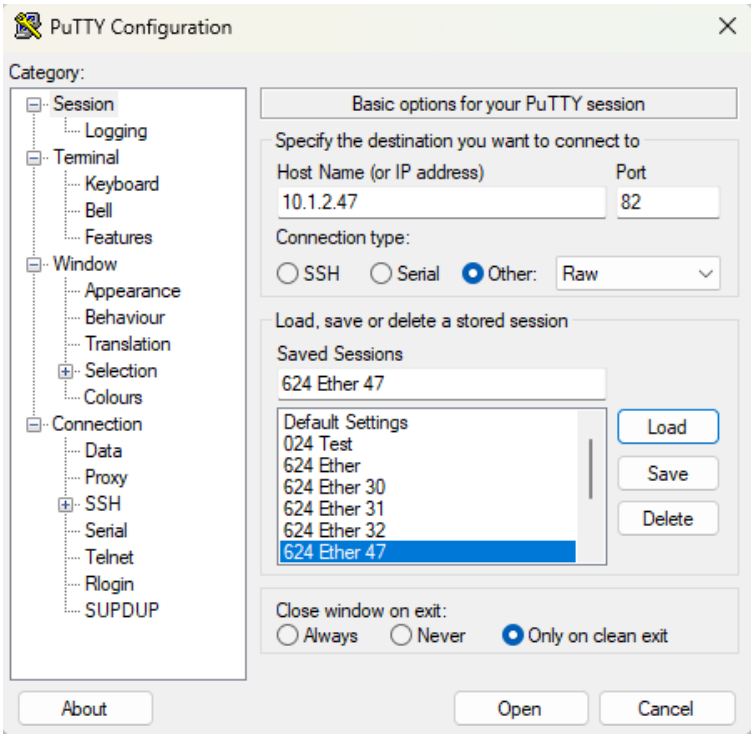

Image 3: PuTTY Configuration – IP address and Port

Under the Terminal section ensure that Local Echo and Local Line Editing are both set to Force On

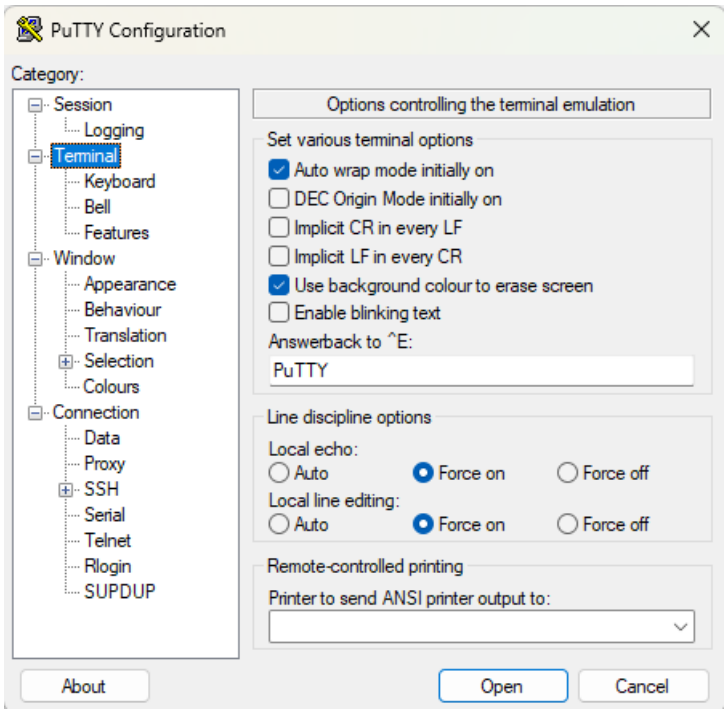

Image 4: PuTTY Configuration - Local echo and Local line editing

A connection with the instrument should now be established.

The connection can be tested by sending the Identity? query. The instrument will return its identity string, e.g. FLANN MICROWAVE, 624PRVA, 123456, V1.0.

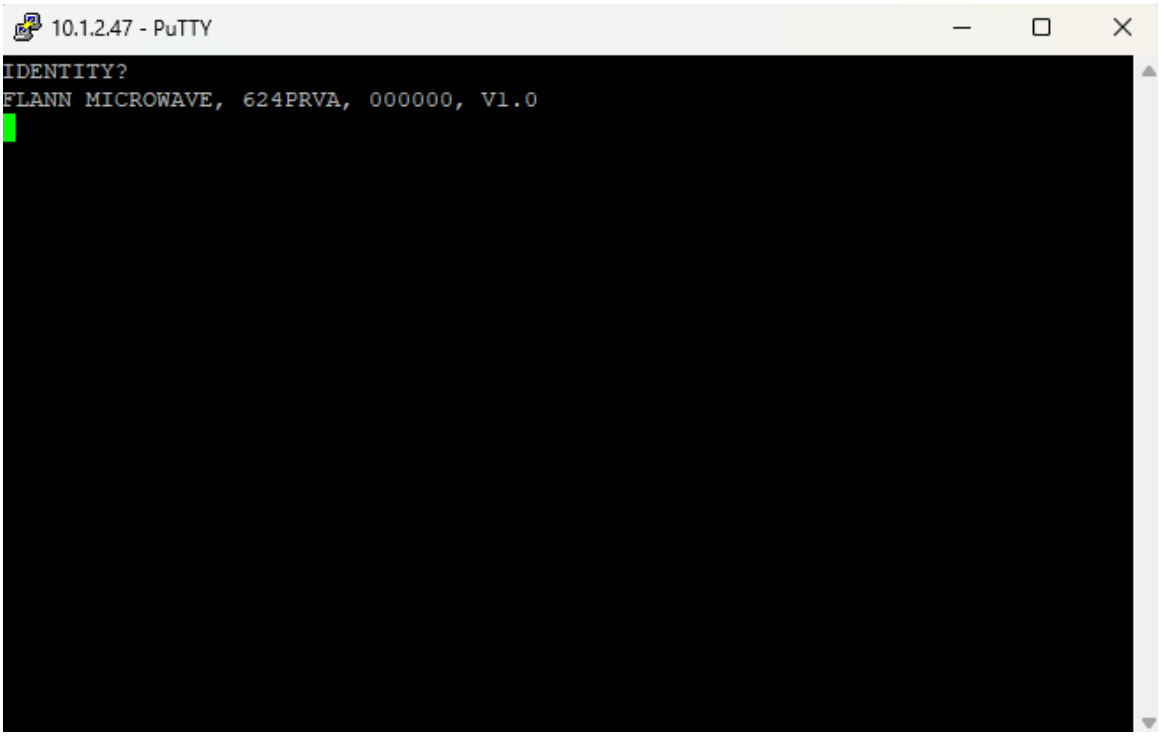

Image 5: PuTTY Configuration – Identity Query and response

Note: The Instrument can now be controlled and will perform functions using the commands illustrated in the Reference Section.

RAW – Python Example

The code below is an example of a Python script that will communicate with the Instrument in the same manner as the section above.

```
import socket

TCP_IP = "10.1.2.31"
TCP_PORT = 82
MESSAGE = b'IDENTITY?\r\n'

try:
   sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   sock.settimeout(10)
   sock.connect((TCP_IP, TCP_PORT))

   count = 0


   count += 1

   if count > 3:
      count = 0

   try:
      sock.sendall(MESSAGE)
      response = b""
      while True:
         try:
            data = sock.recv(32)
            if not data:
               break
            response += data
         except socket.timeout:
            break
      if response:
         print(f"Received response: {response.decode(errors='ignore')}")
      else:
         print("No response received.")
   except Exception as e:
      print(f"Error during send/receive: {e}")

except Exception as e:
   print(f"Connection error: {e}")
finally:
   try:
      sock.shutdown(socket.SHUT_RDWR)
   except Exception:
      pass
   sock.close()
```

Image 6: Example Python script for sending and receiving Identity query

Note: Commands that are sent require a /r/n (Hex 0x0D, 0x0A) terminating the command, this allows the instrument to process commands. Programs like PuTTY will include these automatically.

Note: The Instrument can now be controlled and will perform functions using the commands illustrated in the Reference Section.

REFERENCE SECTIONS

SYSTEM FUNCTIONS AND FEATURES

Power up Procedure

When the instrument is connected to a PoE compatible network the instrument will reference/reset itself to the 50 dB position.
Please note the instrument can be programmed so that after 'power-up' it will return to its most recent attenuation setting (that which it held prior to the power being removed). Refer to the 'HOLD_SET' command.

Definition of STEP and VALUE modes:

Some commands work with regard to the 'mode' that the instrument is currently operating in. The commands VALUE_SET and STEPS_SET set the 'mode' that the instrument is in. Commands like INCR_SET, INCREMENT and DECREMENT operate in the mode set by the most recent VALUE_SET or STEPS_SET command. So, the use of a command VALUE_SET results in the Instrument interpreting the {value} as being in units of dB. Following a STEPS_SET command any INCREMENT will be interpreted as being in Steps.

The instrument is supplied in 'Value' mode, covering the range of 0.0 dB to 50.0 dB, with the smallest incremental value being 0.1 dB. It is possible to position the instrument using motor steps in the range 0 to 2410, giving increased resolution at lower attenuation settings. Refer to the 'STEPS_SET' command and Table 2 below. Note that the steps are from the reference position of 50.0 dB.

Attenuation dB	Steps	Attenuation dB	Steps	Attenuation dB	Steps
50.0	0	33.0	149	16.0	562
49.0	5	32.0	164	15.0	603
48.0	11	31.0	179	14.0	647
47.0	17	30.0	195	13.0	695
46.0	23	29.0	212	12.0	746
45.0	30	28.0	230	11.0	801
44.0	37	27.0	249	10.0	861
43.0	45	26.0	270	9.0	926
42.0	52	25.0	291	8.0	997
41.0	61	24.0	314	7.0	1075
40.0	70	23.0	339	6.0	1162
39.0	79	22.0	365	5.0	1260
38.0	89	21.0	393	4.0	1371
37.0	100	20.0	422	3.0	1501
36.0	111	19.0	454	2.0	1661
35.0	123	18.0	488	1.0	1875
34.0	136	17.0	524	0.0	2410

Table 2: Attenuation – Steps from Reference

Instrument Errors

Instrument errors can be identified by interrogating the Status Byte. Refer to the 'INST_STAT?' command on page 10 and the interpretation list, Table 4, on page 15.

NOTATION

This section details the commands available and the valid operands that may accompany them.

Upper case bold characters represent the program codes, which must appear exactly as listed. Program code commands are not case sensitive, i.e. upper- and lower-case characters are accepted.
Please note the input buffer can accommodate up to 50 bytes maximum. All the native commands (described below) fall within this limit.

Characters enclosed in the {} brackets are qualifiers attached to the root mnemonic. A space may be inserted between it and the root mnemonic. e.g. {ON|OFF} shows that either ON or OFF can be attached to the root mnemonic.

Query Commands

Some commands can be queried to establish their current value; these are identified in Table 3. For instrument state commands the syntax is: append a question mark character (?) instead of the {value} to interrogate the state of the functions. The command requires that no space is present when using the command in this manner.

For example, the 'HOLD_SET' command has three possible applications:

1. **HOLD_SET ON** - Switch on the Hold Set, with or without a space
2. **HOLD_SET OFF** - Switch off the Hold Set, with or without a space
3. **HOLD_SET?** - Query the present state of the Power-on Reset

Each program command must be terminated with \r\n (Hex 0x0a).
For example: 'VALUE_SET45.3\n'

COMMAND SET

Command	Suffix	Action	Query
INCREMENT		Increase setting by stored increment	
DECREMENT		Decrease setting by stored increment	
VALUE_SET	{Value}	Switches to value mode and sets to value	✓
STEPS_SET	{Value}	Switches to steps mode and sets to value	✓
INCR_SET	{Value}	Stores increment in the present operating mode	✓
RESET_INST		Reinitialise the instrument on the active channel	
INST_MODE?		Outputs the current operating mode	✓
PRECISION	{ON/OFF}	Enable or disable the precision setting feature	✓
REC_SETTING		Returns the instrument to the stored setting (see below)	
STORE_VAL	{Value}	Store a setting determined by value (mode dependent)	✓
IDENTITY?		Returns the instrument identity string	✓
INST_STAT?		Returns the Status Register value from 0 to 255	✓
HOLD_SET	{ON/OFF}	Return instrument to last power-off condition	✓

Table 3 – Command Summary

Description of the Commands

VALUE_SET {*value*}
Switches to value mode and sets the microwave instrument to the attenuation ***{value}***
Query: Valid, returns instruments current position in dB
Value 0 to 50.0 (dB). Note: The decimal point would not be required if the desired attenuation is an Integer. VALUE_SET 20 is a valid command.
For example: A valid command would look like VALUE_SET ##.#, or VALUE_SET?

STEPS_SET {*value*}
Switches to Steps mode and sets the microwave instrument to ***{value}***
Query: Valid, returns instruments current position is number of steps
Value 0 to 2410. The {value} is always an integer.
For example: A valid command would look like STEPS_SET ####, or STEPS_SET?

Note: As shown in Table 2, the instrument's step count is zero (0) at 50 dB, increasing the steps value will reduce the instrument's attenuation in dB.

INCR_SET *value*
Sets the stored increment to ***{value}***
Query: Valid, returns stored increment
Value 0 to 50.0 (dB) if in value mode, or 0 to 2410 if in Steps mode. Note: While in VALUE_SET mode INCR_SET does not require the decimal point if the desired attenuation increment is an Integer. INCR_SET 20 is a valid command.
For example: A valid command would look like INCR_SET ##.#, INCR_SET #### or INCR_SET?

INCREMENT
Increase the microwave instrument setting by the stored increment
Query: Invalid

Note: An increment in steps will result in a decrement in dB attenuation

DECREMENT
Decrease microwave instrument setting by the stored increment
Query: Invalid

Note: A decrement in steps will result in an increment in dB attenuation

Note: If an increment or decrement from the instruments current position would take the instrument beyond its limits 0-50 dB or 0-2410, the instrument will remain in its current position and an error will be returned

STORE_VAL {*value*}
Sets the stored setting to ***{value}***
Query: Valid, returns stored setting
Value 0 to 50.0 (dB)
For example: A valid command would look like STORE_VAL ##.#, or STORE_VAL?

Note: This value is retained through a power cycle.

REC_SETTING
Sets the instrument to the stored setting from the STORE_VAL command
Query: Invalid

HOLD_SET {ON|OFF}
On power up returns the microwave instrument to the position when power was removed
Query: Valid, returns 1 for on, or 0 for off
For example: A valid command would look like HOLD_SET ON, or HOLD_SET?

Note: The instrument during the start-up process will move from its current position to 50 dB before returning to the position it was in when power was removed.

INST_MODE?
Outputs the current operating mode
Query: Valid, returns either 0 for VALUE MODE or 1 for STEPS MODE

> **Note: Sending a STEPS_SET [*value*] command while in value mode will change the instrument from value mode to steps mode, the opposite is true of sending a VALUE_SET [*value*] command while in steps mode.**

PRECISION {ON|OFF}
Enables or disables the precision setting feature
Query: Valid, returns 1 for on, or 0 for off
For example: A valid command would look like PRECISION ON, or PRECISION?

> **Note: When ON, this feature gives higher attenuation accuracy and repeatability by always driving to the required position from the same direction. The attenuation value will go higher than the required setting momentarily during positioning from the 0 dB direction. This applies to both Steps and Value modes.**

RESET_INST
Resynchronise the instrument by driving to the reference position of 50.0 dB
Query: Invalid

IDENTITY?
Outputs the identity string, e.g. “FLANN MICROWAVE, 624PRVA, 123456, V1.8”, where 123456 is the instrument serial number, and V1.8 is the firmware version
Query: Valid, returns the identity string of the instrument.

INST_STAT?
Request the value of the status register
Query: Valid, return the status register value.

> **Note: The status response is an 8-bit message, i.e. 00000000. See Status Byte information for what each bit means.**

Examples:

RESET_INST\n	reset the instrument
VALUE_SET? \n	returns a value of 50
VALUE_SET23.4 \n	position to 23.4 dB
VALUE_SET?\n	returns the value 23.4
STEPS_SET453\n	switch to steps mode and position to 453
STEPS_SET?\n	returns the value 453
INCR_SET10\n	store an incremental value of 10 steps
INCREMENT\n	move +10 steps
DECREMENT\n	move -10 steps

A command string must be no more than 50 bytes and terminated by a newline (Hex 0x0a) when in RAW.

STATUS BYTE

Bit	Error
0	EEPROM error – failure to read or write to the EEPROM
1	Out of range request – incorrect value requested
2	Power on – a power-on has occurred since the last read of the register
3	Command error – incorrect syntax in a command line
4	Execution error – failure to achieve setting
5	Not used
6	Error E2 – no encoder output found
7	Error E1 – encoder index not found

Table 4 – Status Byte Interpretation

Note: The value of the Status Register will return to zero after being read.