

---

# **pyTempico**

***Release 1.0.0***

**Tausand Electronics**

**Feb 08, 2024**



# CONTENTS

<b>1</b>	<b>Contents</b>	<b>3</b>
1.1	pyTempico package . . . . .	3
1.2	TempicoDevice class . . . . .	3
1.3	TempicoChannel class . . . . .	11
<b>2</b>	<b>Indices and tables</b>	<b>17</b>
	<b>Python Module Index</b>	<b>19</b>
	<b>Index</b>	<b>21</b>





pyTempico was built to simplify the usage of Tausand Tempico family of time-to-digital converters (TDCs), providing a library aimed to interface these devices using Python coding.



## CONTENTS

## 1.1 pyTempico package

Created on Jan 30 2024

@author: David Guzman at Tausand Electronics

[dguzman@tausand.com](mailto:dguzman@tausand.com)

<https://www.tausand.com>

Core class and methods for PyTempico library.

To use with Tausand Electronics' Time-to-Digital Converters (TDCs) of the family *Tausand Tempico*.

Last edited on 2024-02-08.

## 1.2 TempicoDevice class

**class** `pyTempico.core.TempicoDevice(com_port)`

Tausand Tempico TDC device object.

To create an object of the TempicoDevice class, it is required to send as parameter the desired `com_port`. For example,

```
>>> my_tempico_device_object = pyTempico.TempicoDevice('COM5')
```

To modify or access attributes, **please use methods**. For example,

```
>>> my_tempico_device_object.getIdn()
```

To access attributes of a particular channel, use methods of the TempicoChannel class through attributes `ch1`, `ch2`, `ch3`, `ch4` of this class. For example, to get average cycles on channel 2,

```
>>> my_tempico_device_object.ch2.getAverageCycles()
```

Changing attributes without using methods, do not change the actual parameters in the device.

Accessing attributes without using methods, returns values registered in local memory, that may not be updated.

To begin a measurement and read its results, use methods `measure()` and `fetch()`.

**id\_tempico\_device**

Unique identifier of the TausandDevice object.

**Type**  
int

**ch1**

Object of the TempicoChannel class linked to TDC in channel 1 (input A).

**Type**  
*TempicoChannel*

**ch2**

Object of the TempicoChannel class linked to TDC in channel 2 (input B).

**Type**  
*TempicoChannel*

**ch3**

Object of the TempicoChannel class linked to TDC in channel 3 (input C).

**Type**  
*TempicoChannel*

**ch4**

Object of the TempicoChannel class linked to TDC in channel 4 (input D).

**Type**  
*TempicoChannel*

**device**

Serial port object.

**Type**  
Serial

**idn**

Identification string.

**Type**  
str

**number\_of\_channels**

number of stop inputs of the device.

**Type**  
int

**number\_of\_runs**

Number of measurement runs of the TDCs in TempicoDevice.

**Type**  
int

**port**

Serial port string.

**Type**  
str



**threshold**

Threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

**Type**

float

**close()**

Ends (closes) a connection with a TausandDevice.

It is recommended to close connection at the end of a routine, to free the device's port for future use.

**Parameters**

(none) –

**closeTempico()**

Ends (closes) a connection with a TausandDevice.

Same as method [close\(\)](#).

**Parameters**

(none) –

**convertReadDataToIntList(*data\_string*)**

Converts a string with a read dataset message issued by a TempicoDevice, into an integer 2D-list.

The dataset of a TempicoDevice is in the following format:

```
[[ch,run,start_us,stop_ps1,...,stop_psN],...,[ch,run,start_time_us,stop_ps1,...,
↪stop_psN]]
```

where

- 'ch' indicates the TDC channel,
- 'run' goes from 1 to NumberOfRuns,
- 'start\_us' is the timestamp of start pulse, in microseconds; this value overflows (go back to zero) after  $2^{32}-1$  microseconds, about 4294 seconds,
- 'stop\_ps1' is the measured precision timelapse between start and the 1st stop pulse, in picoseconds,
- 'N' is the NumberOfStops.

Every value in the dataset is an integer.

**Parameters**

**data\_string** (*str*) – dataset message to convert.

**Returns**

dataset message converted.

**Return type**

list(int)

**decrementThresholdVoltage()**

Reduces the threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

To validate the actual threshold voltage applied, method [getThresholdVoltage\(\)](#) should be called.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

(none) –

**fetch()**

Reads the most recent measurement data set from a TempicoDevice.

The dataset of a TempicoDevice is in the following format:

```
[[ch,run,start_us,stop_ps1,...,stop_psN],...,[ch,run,start_time_us,stop_ps1,...,  
↪stop_psN]]
```

where

- 'ch' indicates the TDC channel,
- 'run' goes from 1 to NumberOfRuns,
- 'start\_us' is the timestamp of start pulse, in microseconds; this value overflows (go back to zero) after  $2^{32}-1$  microseconds, about 4294 seconds,
- 'stop\_ps1' is the measured precision timelapse between start and the 1st stop pulse, in picoseconds,
- 'N' is the NumberOfStops.

Every value in the dataset is an integer.

If no measurement has been done, the device may respond with an empty dataset. To make a measurement, method `measure()` must be used.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

(none) –

**Returns**

measured dataset.

**Return type**

list(int)

**getBaudRate()**

Returns the TempicoDevice baud rate.

**Parameters**

(none) –

**Returns**

baud rate.

**Return type**

int

**getFirmware()**

Returns the TempicoDevice firmware version.

**Parameters**

(none) –

**Returns**

firmware version.

**Return type**

str

**getIdn()**

Returns the TempicoDevice identification string.

If the connection is established with the TempicoDevice, this function request the device for the string. If not, the most recent read string is returned.

**Parameters**  
(none) –

**Returns**  
identification string.

**Return type**  
str

**getNumberOfRuns()**

Returns the number of measurement runs of the TDCs in TempicoDevice.

By default, number of runs = 1 (single measurement).

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**  
(none) –

**Returns**  
Number of number of runs.

**Return type**  
integer

**getSettings()**

Reads the current settings form a TempicoDevice.

The response for settings query on a TempicoDevice is in the following format:

```
CH1:ACYC 1;CH1:ENAB 1;CH1:NST 1; . . . ;CH4:STOP:MASK 0;NRUN 1;THR 1.00
```

This function requires that a connection is established with the TempicoDevice.

**Parameters**  
(none) –

**Returns**  
device settings.

**Return type**  
str

**getThresholdVoltage()**

Returns the threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

Start and stop inputs are coupled to 50 ohms.

By default, threshold voltage = 1.00V (recommended for TTL>2.5V).

All inputs are 5V tolerant.

Gate input. This parameter does not have effect on the gate input. Gate input accepts 3.3V TTL and 5V TTL signals.

- When gate is disconnected, system is enabled.

- When gate is connected to 0V, system is disabled.
- When gate is connected to 3.3V/5V, system is enabled.

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**  
(none) –

**Returns**  
start and stop inputs threshold voltage.

**Return type**  
float

#### **incrementThresholdVoltage()**

Increases the threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

To validate the actual threshold voltage applied, method [\*getThresholdVoltage\(\)\*](#) should be called.

This function requires that a connection is established with the TempicoDevice.

**Parameters**  
(none) –

#### **isOpen()**

Returns if a TDC TempicoDevice connection is established (open).

**Parameters**  
(none) –

**Returns**  
True when TempicoDevice connection is open.

**Return type**  
bool

#### **isPendingReadMessage()**

Determines if a pending message is available to be read in a TempicoDevice serial port.

**Parameters**  
(none) –

**Returns**  
True, when a pending message is found.

**Return type**  
bool

#### **measure()**

Begins a measurement sequence and reads its dataset from a TempicoDevice.

The dataset of a TempicoDevice is in the following format:

```
[[ch,run,start_us,stop_ps1,...,stop_psN],...,[ch,run,start_time_us,stop_ps1,...,  
↪stop_psN]]
```

where

- ‘ch’ indicates the TDC channel,
- ‘run’ goes from 1 to NumberOfRuns,

- 'start\_us' is the timestamp of start pulse, in microseconds; this value overflows (go back to zero) after  $2^{32}-1$  microseconds, about 4294 seconds,
- 'stop\_ps1' is the measured precision timelapse between start and the 1st stop pulse, in picoseconds,
- 'N' is the NumberOfStops.

Every value in the dataset is an integer.

If measurement cannot be completed within timeout, the device may respond with an incomplete or empty dataset. In this case, to obtain a complete dataset, the method `fetch()` may be called later.

This function requires that a connection is established with the TempicoDevice.

**Parameters**  
(none) –

**Returns**  
measured dataset.

**Return type**  
list(int)

### `open()`

Establishes (opens) a connection with a TausandDevice.

It is mandatory to establish a connection with this method before to be able to send/to receive data to/from the device.

**Parameters**  
(none) –

### `openTempico()`

Establishes (opens) a connection with a TausandDevice.

Same as method `open()`.

**Parameters**  
(none) –

### `readIdnFromDevice()`

Returns the TempicoDevice identification string, by requesting it to the device.

This function requires that a connection is established with the TempicoDevice. As an alternative, method `getIdn()` may be used.

**Parameters**  
(none) –

**Returns**  
identification string.

**Return type**  
str

### `readMessage()`

Reads pending messages sent by a TempicoDevice from its serial port.

If no message is received, it waits the port timeout, typically 1s.

This function requires that a connection is established with the TempicoDevice.

**Parameters**  
(none) –

**Returns**

read message.

**Return type**

str

**reset()**

Sends a reset command to the TempicoDevice.

Applying a reset clears all the settings of the TempicoDevice and its TempicoChannels to their default values.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

(none) –

**setNumberOfRuns(*number*)**

Modifies the number of measurement runs of the TDCs in TempicoDevice.

By default, number of runs = 1 (single measurement).

This function requires that a connection is established with the TempicoDevice.

**Parameters**

**number** (*int*) – desired number of runs for every TDC. Valid values are from 1 to 1000.

**setThresholdVoltage(*desired\_voltage*)**

Changes the threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

Start and stop inputs are coupled to 50 ohms.

By default, threshold voltage = 1.00V (recommended for TTL>2.5V).

All inputs are 5V tolerant.

Gate input. This parameter does not have effect on the gate input. Gate input accepts 3.3V TTL and 5V TTL signals.

- When gate is disconnected, system is enabled.
- When gate is connected to 0V, system is disabled.
- When gate is connected to 3.3V/5V, system is enabled.

To validate the actual threshold voltage applied, method [getThresholdVoltage\(\)](#) should be called.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

**desired\_voltage** (*float*) – desired start and stop inputs threshold voltage. Valid parameters are MINimum|MAXimum|DOWN|UP or a number from 0.90 to 1.60.

**setThresholdVoltageToMaximum()**

Sets to the maximum valid value the threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

To validate the actual threshold voltage applied, method [getThresholdVoltage\(\)](#) should be called.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

(none) –

**setThresholdVoltageToMinimum()**

Sets to the minimum valid value the threshold voltage on the rising edge of start and stops inputs of TDCs in the TempicoDevice.

To validate the actual threshold voltage applied, method `getThresholdVoltage()` should be called.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

(none) –

**waitAndReadMessage(wait\_time\_ms=1)**

Waits the specified time, and then reads pending messages sent by a TempicoDevice from its serial port, if any.

If no message is received, it does not wait for a port timeout.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

**wait\_time\_ms** (*int*, *optional*) – Waiting time, in milliseconds. Defaults to 1.

**Returns**

read message.

**Return type**

str

**writeMessage(message)**

Writes a message to a TempicoDevice in its serial port.

If a response is expected after writing a message, the `readMessage()` method should be called afterwards to obtain the response.

This function requires that a connection is established with the TempicoDevice.

**Parameters**

**message** (*str*) – message to be sent.

## 1.3 TempicoChannel class

**class pyTempico.core.TempicoChannel(id\_device, ch\_num)**

Single channel on a Tempico Device.

To modify or access attributes, **please use methods**. For example, to get average cycles on channel 2,

```
>>> my_tempico_device_object.ch2.getAverageCycles()
```

Changing attributes without using methods, do not change the actual parameters in the device.

Accessing attributes without using methods, returns values registered in local memory that may not be updated.

**id\_tempico\_channel**

Unique identifier for a TempicoChannel object.

**Type**

int

**id\_tempico\_device**

Identifier of the TausandDevice linked to this TempicoChannel object.

**Type**  
int

**average\_cycles**

Average cycles.

**Type**  
int

**channel\_number**

Number of the channel in the device (1=A, 2=B,...).

**Type**  
int

**enable**

True when the channel is enabled.

**Type**  
bool

**mode**

Measurement mode. 1|2.

- 1: Short measurement range. Start-stop times from 12ns to 500ns.
- 2: Large measurement range. Start-stop times from 125ns to 4ms.

**Type**  
int

**number\_of\_stops**

Number of stop pulses expected after a start pulse arrives. 1..5.

**Type**  
int

**parent\_tempico\_device**

Pointer reference to parent object of TempicoDevice() class.

**Type**  
*TempicoDevice*

**start\_edge**

Edge type on the start pulse used to begin timing. RISE|FALL.

**Type**  
str

**stop\_edge**

Edge type on the stop pulses used to end timing. RISE|FALL.

**Type**  
str



**stop\_mask**

Time that stop pulses are ignored after receiving a start pulse on the TDC. Value in microseconds. 0..4000.

**Type**

int

**disableChannel()**

Disables a TDC TempicoChannel.

By default, channels are enabled.

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

To validate the status of the channel, method *isEnabled()* may be used.

**Parameters**

(none) –

**enableChannel()**

Enables a TDC TempicoChannel.

By default, channels are enabled.

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

To validate the status of the channel, method *isEnabled()* may be used.

**Parameters**

(none) –

**getAverageCycles()**

Returns the average cycles of the TDC TempicoChannel.

By default, average cycles = 1 (no multi-cycle averaging).

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**

(none) –

**Returns**

Number of average cycles.

**Return type**

integer

**getMode()**

Returns the measurement mode of the TDC TempicoChannel.

By default, mode = 1.

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**

(none) –

**Returns**

Mode. Possible values are,

- 1: Short measurement range. Start-stop times from 12ns to 500ns.
- 2: Large measurement range. Start-stop times from 125ns to 4ms.

**Return type**

integer

**getNumberOfStops()**

Returns the expected number of stop pulses of the TDC TempicoChannel.

By default, number of stops = 1 (single start -> single stop).

The TDC must receive all the expected number of stops to register them as a valid measurement; otherwise, the measurements are discarded.

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**

(none) –

**Returns**

Number of stops.

**Return type**

integer

**getStartEdge()**

Returns the edge type used on start pulses of the TDC TempicoChannel.

By default, start edge = 'RISE'.

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**

(none) –

**Returns**

start edge type. Possible values are,

- 'RISE': TDC timing starts on a rising edge of the start pulse.
- 'FALL': TDC timing starts on a falling edge of the start pulse.

**Return type**

string

**getStopEdge()**

Returns the edge type used on stop pulses of the TDC TempicoChannel.

By default, stop edge = 'RISE'.

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**

(none) –

**Returns**

stop edge type. Possible values are,

- 'RISE': TDC timing ends on a rising edge of the stop pulse.
- 'FALL': TDC timing ends on a falling edge of the stop pulse.

**Return type**

string

**getStopMask()**

Returns the time that stop pulses are ignored after receiving a start pulse on the TDC TempicoChannel. In microseconds.

By default, stop mask = 0 (no masking).

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**  
(none) –

**Returns**  
stop mask time, in microseconds.

**Return type**  
integer

**isEnabled()**

Returns if a TDC TempicoChannel is enabled.

By default, channels are enabled.

If the connection is established with the TempicoDevice, this function request the device for the value. If not, the most recent value is returned.

**Parameters**  
(none) –

**Returns**  
True, when TDC channel is enabled.

**Return type**  
bool

**setAverageCycles(number)**

Modifies the average cycles of the TDC TempicoChannel.

By default, average cycles = 1 (no multi-cycle averaging).

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

**Parameters**  
**number** (*int*) – desired average cycles for the TDC. Valid values are 1|2|4|8|16|32|64|128.

**setMode(number)**

Modifies the measurement mode of the TDC TempicoChannel.

By default, mode = 1. Possible values are,

- 1: Short measurement range. Start-stop times from 12ns to 500ns.
- 2: Large measurement range. Start-stop times from 125ns to 4ms.

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

**Parameters**  
**number** (*int*) – desired measurement mode for the TDC. Valid values are 1 or 2.

**setNumberOfStops(number)**

Modifies the expected number of stop pulses of the TDC TempicoChannel.

By default, number of stops = 1 (single start -> single stop).

The TDC must receive all the expected number of stops to register them as a valid measurement; otherwise, the measurements are discarded. For extending the valid time range, consider using measurement mode 2.

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

**Parameters**

**number** (*int*) – desired number of stops for the TDC. Valid values are from 1 to 5.

**setStartEdge**(*edge\_type*)

Sets the edge type used on start pulses of the TDC TempicoChannel.

By default, start edge = 'RISE'. Possible values are,

- 'RISE': TDC timing starts on a rising edge of the start pulse.
- 'FALL': TDC timing starts on a falling edge of the start pulse.

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

**Parameters**

**edge\_type** (*str*) – desired start edge type for the TDC. Valid values are 'RISE', 1, 'FALL', 0.

**setStopEdge**(*edge\_type*)

Sets the edge type used on stop pulses of the TDC TempicoChannel.

By default, stop edge = 'RISE'. Possible values are,

- 'RISE': TDC timing ends on a rising edge of the stop pulse.
- 'FALL': TDC timing ends on a falling edge of the stop pulse.

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

**Parameters**

**edge\_type** (*str*) – desired stop edge type for the TDC. Valid values are 'RISE', 1, 'FALL', 0.

**setStopMask**(*stop\_mask\_in\_us*)

Modifies the time that stop pulses are ignored after receiving a start pulse on the TDC TempicoChannel.

By default, stop mask = 0 (no masking).

This function requires that a connection is established with the TempicoDevice of the TempicoChannel.

**Parameters**

**stop\_mask\_in\_us** (*int*) – desired stop mask for the TDC, in microseconds. Valid values are from 0 to 4000.

## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### p

`pyTempico.core`, 3





## A

`average_cycles` (*pyTempico.core.TempicoChannel* attribute), 12

## C

`ch1` (*pyTempico.core.TempicoDevice* attribute), 4  
`ch2` (*pyTempico.core.TempicoDevice* attribute), 4  
`ch3` (*pyTempico.core.TempicoDevice* attribute), 4  
`ch4` (*pyTempico.core.TempicoDevice* attribute), 4  
`channel_number` (*pyTempico.core.TempicoChannel* attribute), 12  
`close()` (*pyTempico.core.TempicoDevice* method), 5  
`closeTempico()` (*pyTempico.core.TempicoDevice* method), 5  
`convertReadDataToIntList()` (*pyTempico.core.TempicoDevice* method), 5

## D

`decrementThresholdVoltage()` (*pyTempico.core.TempicoDevice* method), 5  
`device` (*pyTempico.core.TempicoDevice* attribute), 4  
`disableChannel()` (*pyTempico.core.TempicoChannel* method), 13

## E

`enable` (*pyTempico.core.TempicoChannel* attribute), 12  
`enableChannel()` (*pyTempico.core.TempicoChannel* method), 13

## F

`fetch()` (*pyTempico.core.TempicoDevice* method), 5

## G

`getAverageCycles()` (*pyTempico.core.TempicoChannel* method), 13  
`getBaudRate()` (*pyTempico.core.TempicoDevice* method), 6  
`getFirmware()` (*pyTempico.core.TempicoDevice* method), 6  
`getIdn()` (*pyTempico.core.TempicoDevice* method), 6  
`getMode()` (*pyTempico.core.TempicoChannel* method), 13

`getNumberOfRuns()` (*pyTempico.core.TempicoDevice* method), 7  
`getNumberOfStops()` (*pyTempico.core.TempicoChannel* method), 14  
`getSettings()` (*pyTempico.core.TempicoDevice* method), 7  
`getStartEdge()` (*pyTempico.core.TempicoChannel* method), 14  
`getStopEdge()` (*pyTempico.core.TempicoChannel* method), 14  
`getStopMask()` (*pyTempico.core.TempicoChannel* method), 15  
`getThresholdVoltage()` (*pyTempico.core.TempicoDevice* method), 7

## I

`id_tempico_channel` (*pyTempico.core.TempicoChannel* attribute), 11  
`id_tempico_device` (*pyTempico.core.TempicoChannel* attribute), 11  
`id_tempico_device` (*pyTempico.core.TempicoDevice* attribute), 3  
`idn` (*pyTempico.core.TempicoDevice* attribute), 4  
`incrementThresholdVoltage()` (*pyTempico.core.TempicoDevice* method), 8  
`isEnabled()` (*pyTempico.core.TempicoChannel* method), 15  
`isOpen()` (*pyTempico.core.TempicoDevice* method), 8  
`isPendingReadMessage()` (*pyTempico.core.TempicoDevice* method), 8

## M

`measure()` (*pyTempico.core.TempicoDevice* method), 8  
`mode` (*pyTempico.core.TempicoChannel* attribute), 12  
`module`  
`pyTempico.core`, 3

## N

`number_of_channels` (*pyTempico.core.TempicoDevice* attribute), 4  
`number_of_runs` (*pyTempico.core.TempicoDevice* attribute), 4

`number_of_stops` (*pyTempico.core.TempicoChannel* attribute), 12

## O

`open()` (*pyTempico.core.TempicoDevice* method), 9

`openTempico()` (*pyTempico.core.TempicoDevice* method), 9

## P

`parent_tempico_device` (*pyTempico.core.TempicoChannel* attribute), 12

`port` (*pyTempico.core.TempicoDevice* attribute), 4

`pyTempico.core`  
module, 3

## R

`readIdnFromDevice()` (*pyTempico.core.TempicoDevice* method), 9

`readMessage()` (*pyTempico.core.TempicoDevice* method), 9

`reset()` (*pyTempico.core.TempicoDevice* method), 10

## S

`setAverageCycles()` (*pyTempico.core.TempicoChannel* method), 15

`setMode()` (*pyTempico.core.TempicoChannel* method), 15

`setNumberOfRuns()` (*pyTempico.core.TempicoDevice* method), 10

`setNumberOfStops()` (*pyTempico.core.TempicoChannel* method), 15

`setStartEdge()` (*pyTempico.core.TempicoChannel* method), 16

`setStopEdge()` (*pyTempico.core.TempicoChannel* method), 16

`setStopMask()` (*pyTempico.core.TempicoChannel* method), 16

`setThresholdVoltage()` (*pyTempico.core.TempicoDevice* method), 10

`setThresholdVoltageToMaximum()` (*pyTempico.core.TempicoDevice* method), 10

`setThresholdVoltageToMinimum()` (*pyTempico.core.TempicoDevice* method), 10

`start_edge` (*pyTempico.core.TempicoChannel* attribute), 12

`stop_edge` (*pyTempico.core.TempicoChannel* attribute), 12

`stop_mask` (*pyTempico.core.TempicoChannel* attribute), 12

## T

`TempicoChannel` (class in *pyTempico.core*), 11

`TempicoDevice` (class in *pyTempico.core*), 3

`threshold` (*pyTempico.core.TempicoDevice* attribute), 4

## W

`waitAndReadMessage()` (*pyTempico.core.TempicoDevice* method), 11

`writeMessage()` (*pyTempico.core.TempicoDevice* method), 11