PyAbacus Documentation

Release 1.0.6

Tausand Electronica

Jan 03, 2019

CONTENTS

Tausand

pyAbacus was build to simplify the usage of Tausands tools.

CHAPTER

ONE

CONTENTS

1.1 pyAbacus.core

class pyAbacus.core.**AbacusSerial**(*port*, *bounce_timeout=40*)

Builds a serial port from pyserial.

findIdn()

flush()

getIdn()

getNChannels()

readSerial()

testDevice()

writeSerial(*command*, *address*, *data_16o32*)

class pyAbacus.core.**CountersValues**(*n_channels*)

getCountersID()

getNumericAddresses()

getTimeLeft()

getValue(*channel*)

getValues(*channels*)

getValuesFormatted(*channels*)

setCountersID(*id*)

setTimeLeft(*time*)

setValueFromArray(*address*, *value*)

time_left = None

in ms

class pyAbacus.core.**Settings2Ch**

getAddressAndValue(*timer*)

getSetting(*timer*)

getSettingStr(*timer*)

setSetting(*setting*, *value*)

class pyAbacus.core.**Settings48Ch**

4 and 8 channel devices use as time base a second. Nevertheless 2 channel uses ns for all timers with the exception of the sampling time (ms).

exponentRepresentationToValue(*c*, *e*)

exponentsToBits(*c*, *e*)

fromBitsToValue(*bits*)

getAddressAndValue(*timer*)

getChannels()

getSetting(*timer*)

For all timers: returns nanoseconds, for sampling returns ms.

getSettingStr(*timer*)

initAddreses()

setSetting(*setting*, *value*)

For all timers: value is in nanoseconds, for sampling in ms.

valueToExponentRepresentation(*number*)

class pyAbacus.core.**Settings4Ch**

4 and 8 channel devices use as time base a second. Nevertheless 2 channel uses ns for all timers with the exception of the sampling time (ms).

class pyAbacus.core.**Settings8Ch**

4 and 8 channel devices use as time base a second. Nevertheless 2 channel uses ns for all timers with the exception of the sampling time (ms).

class pyAbacus.core.**Stream**(*abacus_port*, *counters*, *output_function=<built-in function print>*)

setCounters(*counters*)

start()

stop()

pyAbacus.core.**close**(*abacus_port*)

pyAbacus.core.**dataArraysToCounters**(*abacus_port*, *addresses*, *data*)

pyAbacus.core.**dataArraysToSettings**(*abacus_port*, *addresses*, *data*)

pyAbacus.core.**dataStreamToDataArrays**(*input_string*)

pyAbacus.core.**findDevices**(*print_on=True*)

pyAbacus.core.**getAllCounters**(*abacus_port*)

pyAbacus.core.**getAllSettings**(*abacus_port*)

pyAbacus.core.**getChannelsFromName**(*name*)

pyAbacus.core.**getCountersID**(*abacus_port*)

pyAbacus.core.**getFollowingCounters**(*abacus_port*, *counters*)

pyAbacus.core.**getIdn**(*abacus_port*)

pyAbacus.core.**getSetting**(*abacus_port*, *setting*) pyAbacus.core.**getTimeLeft**(*abacus_port*)

pyAbacus.core.**open**(*abacus_port*)

pyAbacus.core.**readSerial**(*abacus_port*)

pyAbacus.core.**renameDuplicates**(*old*)

pyAbacus.core.**setAllSettings**(*abacus_port*, *new_settings*)

pyAbacus.core.**setSetting**(*abacus_port*, *setting*, *value*)

pyAbacus.core.**writeSerial**(*abacus_port*, *command*, *address*, *data_16o32*)

1.2 pyAbacus.exceptions

exception pyAbacus.exceptions.**AbacusError**(*message=''*)
    An unexpected error ocurred.

exception pyAbacus.exceptions.**BaseError**(*message*)

exception pyAbacus.exceptions.**CheckSumError**
    An error ocurred while doing check sum.

exception pyAbacus.exceptions.**InvalidValueError**(*message=''*)
    The selected value is not valid

exception pyAbacus.exceptions.**TimeOutError**(*message=''*)
    A time out error ocurred

1.3 pyAbacus.constants

pyAbacus.constants.**ADDRESS_DIRECTORY_2CH = {'coincidence_window_ms': 22, 'coincidence_win**
    Memory addresses

pyAbacus.constants.**BAUDRATE = 115200**
    Default baudrate for the serial port communication

pyAbacus.constants.**BOUNCE_TIMEOUT = 40**
    Number of times a specific transmition is tried

pyAbacus.constants.**COINCIDENCE_WINDOW_DEFAULT_VALUE = 5**
    Default coincidence window time value (ns).

pyAbacus.constants.**COINCIDENCE_WINDOW_MAXIMUM_VALUE = 50000**
    Maximum coincidence window time value (ns).

pyAbacus.constants.**COINCIDENCE_WINDOW_MINIMUM_VALUE = 5**
    Minimum coincidence window time value (ns).

pyAbacus.constants.**COINCIDENCE_WINDOW_STEP_VALUE = 5**
    Increase ratio on the coincidence window time value (ns).

pyAbacus.constants.**COUNTERS_VALUES = {}**
    Global counters values variable

pyAbacus.constants.**CURRENT_OS = 'linux'**
    Current operative system

`pyAbacus.constants.`**`DELAY_DEFAULT_VALUE = 100`**
: Default delay time value (ns).

`pyAbacus.constants.`**`DELAY_MAXIMUM_VALUE = 100`**
: Maximum delay time value (ns).

`pyAbacus.constants.`**`DELAY_MINIMUM_VALUE = 0`**
: Minimum delay time value (ns).

`pyAbacus.constants.`**`DELAY_STEP_VALUE = 5`**
: Increase ratio on the delay time value (ns).

`pyAbacus.constants.`**`END_COMMUNICATION = 4`**
: End of message

`pyAbacus.constants.`**`MAXIMUM_WRITING_TRIES = 20`**
: Number of tries done to write a value

`pyAbacus.constants.`**`READ_VALUE = 14`**
: Reading operation signal

`pyAbacus.constants.`**`SAMPLING_DEFAULT_VALUE = 100`**
: Default sampling time value (ms)

`pyAbacus.constants.`**`SAMPLING_VALUES = [1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000,`**
: From (1, 2, 5) ms to 1000 s

`pyAbacus.constants.`**`SETTINGS = {}`**
: Global settings variable

`pyAbacus.constants.`**`SLEEP_DEFAULT_VALUE = 25`**
: Default sleep time value (ns).

`pyAbacus.constants.`**`SLEEP_MAXIMUM_VALUE = 100`**
: Maximum sleep time value (ns).

`pyAbacus.constants.`**`SLEEP_MINIMUM_VALUE = 0`**
: Minimum sleep time value (ns).

`pyAbacus.constants.`**`SLEEP_STEP_VALUE = 5`**
: Increase ratio on the sleep time value (ns).

`pyAbacus.constants.`**`START_COMMUNICATION = 2`**
: Begin message signal

`pyAbacus.constants.`**`TIMEOUT = 0.5`**
: Maximum time without answer from the serial port

`pyAbacus.constants.`**`WRITE_VALUE = 15`**
: Writing operation signal

CHAPTER TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX