
python_ics Documentation

Release 4.2

David Rebbe

Aug 19, 2019

Contents

1	Versioning Information	3
1.1	v802	3
1.2	v803	3
1.3	v900	3
2	Installation on Windows	5
2.1	Building from source	5
2.2	Intrepid icsneo40 Library	5
3	Installation on Linux	7
3.1	Fedora Dependencies (FC28)	7
3.2	Debian/Ubuntu Dependencies	7
3.3	Others (Required dependencies)	7
3.4	Installation	7
3.5	Intrepid libicsneoapi.so Library	8
4	Getting Started	9
5	Module Function List	11
6	Module Documentation	19
7	Module Variables	71
	Python Module Index	83
	Index	85

Intrepid Control Systems, Inc. open source Python module for interfacing to Intrepid hardware. Basic knowledge of using and installing Python modules is assumed. Please see <https://docs.python.org/3/installing/index.html> for documentation on how to install Python modules.

Versioning Information

Minor differences can occur between different icsnVC40.h versions. These differences are usually just structures and constant differences. Below is a list of how the python_ics version correlates to the icsnVC40.h version:

1.1 v802

```
pip install 'python_ics>=2.0,<3.0' --force-reinstall
```

Note: Refer to platform specific installation if not on Windows

1.2 v803

```
pip install 'python_ics>=3.0,<4.0' --force-reinstall
```

Note: Refer to platform specific installation if not on Windows

1.3 v900

```
pip install 'python_ics>=4.0,<5.0' --force-reinstall
```

Note: Refer to platform specific installation if not on Windows

Installation on Windows

PyPi provides binary packages for Windows. You can simply install the `python_ics` module by running the following command:

```
pip install python_ics
```

Note: `pip.exe` is usually located under the `Scripts` directory under the Python installation directory.

2.1 Building from source

Building from source on windows is not usually need so it won't really be covered here in detail. As a starting point you'll need to match the compiler version used to build the official Python binaries (MSVC). If the build environment is setup correctly, you should be able to run `python setup.py build` like usual.

2.2 Intrepid icsneo40 Library

`python_ics` module looks for `icsneo40.dll` in the normal windows DLL search paths. The module will throw an exception if its not found.

python_ics does not provide binaries for linux distributions so we will have to compile from source. This can be easily achieved by utilizing Python's PIP. First we need to make sure we have some base packages installed.

3.1 Fedora Dependencies (FC28)

```
sudo dnf install redhat-rpm-config gcc g++ python3-devel
```

3.2 Debian/Ubuntu Dependencies

```
sudo apt install build-essential python-dev
```

3.3 Others (Required dependencies)

- GCC
- G++
- Python Development packages (We Need to link to Python.h)

3.4 Installation

After dependencies are installed we can run the following pip command:

```
pip install python_ics
```

Note: A lot of distributions have Python 2 and 3 installed side by side. As of this writing without a version suffix the commands still default to version 2 of the Python binaries. In order to utilize the Python 3 binaries you must append a 3 after the binary names (python3 and pip3 instead of just python and pip).

3.5 Intrepid libicsneoapi.so Library

Please see <https://github.com/intrepidcs/icsneoapi> for more details.

CHAPTER 4

Getting Started

Please see https://github.com/intrepidcs/python_ics/tree/master/examples for simple examples on how to use this module. Most function documentation has a simple example on how its intended to be used. Every function was designed to be as close as possible to its C counterpart unless it was deemed to make the function more pythonic in nature.

For those experienced with the C API `ics.open_device (icsneoOpenNeoDevice())` behavior has been changed the most (no parameters makes it auto utilize `ics.find_devices (icsneoFindNeoDevices())` and open the first device). Also since python is a object oriented language the module utilizes this and auto cleans up device handles when going out of scope so there is usually no need to call `ics.close_device()`.

CHAPTER 5

Module Function List

<code>ics.close_device</code>	Closes the device.
<code>ics.coremini_clear</code>	Clears the CoreMini into the device.
<code>ics.coremini_get_fblock_status</code>	Gets the status of a Coremini Function Block at <i>index</i> on <i>device</i> .
<code>ics.coremini_get_status</code>	Gets the status of the CoreMini in the device.
<code>ics.coremini_load</code>	Loads the CoreMini into the device.
<code>ics.coremini_read_app_signal</code>	Gets the value of a Coremini application signal at <i>index</i> on <i>device</i> .
<code>ics.coremini_read_rx_message</code>	Gets the value of a Coremini Message at <i>index</i> on <i>device</i> .
<code>ics.coremini_read_tx_message</code>	Gets the value of a Coremini Message at <i>index</i> on <i>device</i> .
<code>ics.coremini_start</code>	Starts the CoreMini into the device.
<code>ics.coremini_start_fblock</code>	Starts a Coremini Function Block at <i>index</i> on <i>device</i> .
<code>ics.coremini_stop</code>	Stops the CoreMini into the device.
<code>ics.coremini_stop_fblock</code>	Stops a Coremini Function Block at <i>index</i> on <i>device</i> .
<code>ics.coremini_write_app_signal</code>	Sets the value of a Coremini application signal at <i>index</i> on <i>device</i> .
<code>ics.coremini_write_rx_message</code>	TODO
<code>ics.coremini_write_tx_message</code>	TODO
<code>ics.create_neovi_radio_message</code>	Python API only.
<code>ics.find_devices</code>	Finds all connected devices and returns a tuple of <code>ics.NeoDevice</code> for use in <code>ics.open_device()</code>
<code>ics.firmware_update_required</code>	Determines if the device firmware needs flashing.
<code>ics.force_firmware_update</code>	Forces the device to flash firmware.
<code>ics.get_active_vnet_channel</code>	Gets active vnet channel for the device.
<code>ics.get_backup_power_enabled</code>	Returns the device backup power enabled for the device.
<code>ics.get_backup_power_ready</code>	Returns the device backup power is ready for the device.
<code>ics.get_device_settings</code>	Gets the settings in the device.
<code>ics.get_device_status</code>	Returns the device status.

Continued on next page

Table 1 – continued from previous page

<code>ics.get_dll_firmware_info</code>	Returns the DLL firmware info for the device.
<code>ics.get_dll_version</code>	Gets the DLL version.
<code>ics.get_error_messages</code>	Gets the error message(s) on the device.
<code>ics.get_hw_firmware_info</code>	Returns the device firmware info for the device.
<code>ics.get_last_api_error</code>	Gets the error message from the last API call.
<code>ics.get_library_path</code>	
<code>ics.get_messages</code>	Gets the message(s) on the device.
<code>ics.get_performance_parameters</code>	Gets the Performance Parameters on <i>device</i> .
<code>ics.get_rtc</code>	Gets the Real-Time Clock of the device.
<code>ics.get_script_status</code>	Accepts a <code>ics.NeoDevice</code> , exception on error.
<code>ics.get_serial_number</code>	Gets the serial number out of the device.
<code>ics.get_timestamp_for_msg</code>	Calculates the timestamp for a message.
<code>ics.iso15765_disable_networks</code>	Disables ISO15765 networks.
<code>ics.iso15765_enable_networks</code>	Enables ISO15765 networks.
<code>ics.iso15765_receive_message</code>	Setup rx ISO15765 Message.
<code>ics.iso15765_transmit_message</code>	Transmits an ISO15765 Message.
<code>ics.load_default_settings</code>	Load the default settings in the device.
<code>ics.open_device</code>	Opens the device.
<code>ics.override_library_name</code>	Overrides the default search for loading the <code>icsneo40</code> library
<code>ics.read_sdcard</code>	<code>icsneoReadSDCard()</code> , Accepts a <code>ics.NeoDevice</code> and sector index.
<code>ics.request_enter_sleep_mode</code>	Signal neoVI to immediate go to sleep.
<code>ics.set_active_vnet_channel</code>	Sets active vnet channel for the device.
<code>ics.set_backup_power_enabled</code>	Sets the device backup power enabled for the device.
<code>ics.set_bit_rate</code>	Specifies bit rate setting.
<code>ics.set_bit_rate_ex</code>	Sets the bitrate for a given Network ID on the device with extended options.
<code>ics.set_context</code>	Sets the “context” of how <code>icsneoFindNeoDevices(Ex)</code> and <code>icsneoOpenNeoDevice(Ex)</code> function.
<code>ics.set_device_settings</code>	Sets the settings in the device.
<code>ics.set_fd_bit_rate</code>	Sets the FD bitrate for a given Network ID on the device..
<code>ics.set_reflash_callback</code>	Sets the reflash display callback.
<code>ics.set_rtc</code>	Sets the Real-Time Clock of the device.
<code>ics.transmit_messages</code>	Transmits message(s) on the device.
<code>ics.validate_hobject</code>	Validates the handle is valid for a <i>device</i> .
<code>ics.write_sdcard</code>	<code>icsneoWriteSDCard()</code> , Accepts a <code>ics.NeoDevice</code> , sector index, and a bytearray of 512 bytes.
<code>ics.ClosePort</code>	
Note: Compatibility Function	
<code>ics.FindNeoDevices</code>	
Note: Compatibility Function	

Continued on next page

Table 1 – continued from previous page

<i>ics.GetDLLVersion</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetErrorMessages</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetHWFirmwareInfo</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetLastAPIError</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetMessages</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetPerformanceParameters</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetRTC</i>	Note: Compatibility Function
<hr/>	
<i>ics.GetSerialNumber</i>	Note: Compatibility Function
<hr/>	
<i>ics.OpenNeoDevice</i>	Note: Compatibility Function
<hr/>	
<i>ics.RequestEnterSleepMode</i>	Note: Compatibility Function
<hr/>	

Continued on next page

Table 1 – continued from previous page

<i>ics.ScriptClear</i>	Note: Compatibility Function
<i>ics.ScriptGetFBlockStatus</i>	Note: Compatibility Function
<i>ics.ScriptGetScriptStatus</i>	Note: Compatibility Function
<i>ics.ScriptLoad</i>	Note: Compatibility Function
<i>ics.ScriptReadAppSignal</i>	Note: Compatibility Function
<i>ics.ScriptReadRxMessage</i>	Note: Compatibility Function
<i>ics.ScriptReadTxMessage</i>	Note: Compatibility Function
<i>ics.ScriptStart</i>	Note: Compatibility Function
<i>ics.ScriptStartFBlock</i>	Note: Compatibility Function
<i>ics.ScriptStop</i>	Note: Compatibility Function

Continued on next page

Table 1 – continued from previous page

<i>ics.ScriptStopFBlock</i>	Note: Compatibility Function
<i>ics.ScriptWriteAppSignal</i>	Note: Compatibility Function
<i>ics.ScriptWriteRxMessage</i>	Note: Compatibility Function
<i>ics.ScriptWriteTxMessage</i>	Note: Compatibility Function
<i>ics.SetRTC</i>	Note: Compatibility Function
<i>ics.SetReflashDisplayCallback</i>	Note: Compatibility Function
<i>ics.TxMessages</i>	Note: Compatibility Function
<i>ics.ValidateHObject</i>	Note: Compatibility Function
<i>ics.base36enc</i>	Converts a decimal serial number to base36.
<i>ics.icsneoFirmwareUpdateRequired</i>	Note: Compatibility Function
<i>ics.icsneoForceFirmwareUpdate</i>	Note: Compatibility Function

Continued on next page

Table 1 – continued from previous page

<i>ics.icsneoGetActiveVNETChannel</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetBackupPowerEnabled</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetBackupPowerReady</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetDLLFirmwareInfo</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetDeviceStatus</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetFireSettings</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetTimeStampForMsg</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoGetVCAN3Settings</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoISO15765_DisableNetworks</i>	Note: Compatibility Function
<hr/>	
<i>ics.icsneoISO15765_EnableNetworks</i>	Note: Compatibility Function
<hr/>	

Continued on next page

Table 1 – continued from previous page

<i>ics.icsneoISO15765_ReceiveMessage</i>	Note: Compatibility Function
<i>ics.icsneoISO15765_TransmitMessage</i>	Note: Compatibility Function
<i>ics.icsneoLoadDefaultSettings</i>	Note: Compatibility Function
<i>ics.icsneoReadSDCard</i>	Note: Compatibility Function
<i>ics.icsneoScriptGetScriptStatusEx</i>	Note: Compatibility Function
<i>ics.icsneoSetActiveVNETChannel</i>	Note: Compatibility Function
<i>ics.icsneoSetBackupPowerEnabled</i>	Note: Compatibility Function
<i>ics.icsneoSetBitRate</i>	Note: Compatibility Function
<i>ics.icsneoSetBitRateEx</i>	Note: Compatibility Function
<i>ics.icsneoSetContext</i>	Note: Compatibility Function

Continued on next page

Table 1 – continued from previous page

ics.icsneoSetFDBitRate

Note: Compatibility Function

ics.icsneoSetFireSettings

Note: Compatibility Function

ics.icsneoSetVCAN3Settings

Note: Compatibility Function

ics.icsneoWriteSDCard

Note: Compatibility Function

Module Documentation

Python C Code module for interfacing to the icsneo40 dynamic library. Code tries to respect PEP 8 (<http://python.org/dev/peps/pep-0008>). Function naming convention does not follow the tradition c style icsneo40 naming convention as pyics module name acts as the namespace (icsneo portion of the function) and function names are suppose to be lowercase with underscores instead of mixedCase like icsneo API.

C API can be mimiced almost identically by doing the following:

```
>>> import ics as icsneo
>>> devices = icsneo.FindNeoDevices()
>>> for device in devices:
...     print(device.Name, device.SerialNumber)
...
neoVI FIRE 59886
```

Recommended *Python* way by doing the following:

```
>>> import ics
>>> devices = ics.find_devices()
>>> for device in devices:
...     print(device.Name, device.SerialNumber)
...
neoVI FIRE 59886
```

It should be noted that *ics.NeoDevice* is used a little bit differently than the C API. *ics.NeoDevice* contains two extra members:

ics.NeoDevice.AutoHandleClose and *ics.NeoDevice._Handle*

The handle normally returned from *icsneoOpenNeoDevice()* is stored inside *_Handle* and setting *AutoHandleClose* to True (Default) will automatically close the handle when the *ics.NeoDevice* goes out of scope.

Installation:

```
pip install python_ics
```

<https://pypi.python.org/pypi/python-ics>

exception `ics.ArgumentError`

Bases: `Exception`

exception `ics.RuntimeError`

Bases: `Exception`

class `ics.ApiFirmwareInfo`

Bases: `object`

ApiFirmwareInfo object

iAppMajor

iAppMinor

iBoardRevMajor

iBoardRevMinor

iBootLoaderVersionMajor

iBootLoaderVersionMinor

iMainFirmChkSum

iMainFirmDateDay

iMainFirmDateHour

iMainFirmDateMin

iMainFirmDateMonth

iMainFirmDateSecond

iMainFirmDateYear

iMainVnetHWrevMajor

iMainVnetHWrevMinor

iMainVnetSRAMSize

iManufactureDay

iManufactureMonth

iManufactureYear

iType

class `ics.CanFdSettings`

Bases: `object`

CanFdSettings object

FDBRP

FDBaudrate

FDMode

FDTqProp

FDTqSeg1

FDTqSeg2

FDTqSync

class ics.CanSettings

Bases: object

CanSettings object

BRP

Baudrate

The bit rate of a CAN channel can be selected from a list of common bit rates Write the correct enumeration for the desired bit rate and ensure that SetBaudrate is 1(auto)

Mode

CAN controller mode when the neoVI device goes online or runs a CoreMini script. Normal=0 Disabled=1 Listen Only=3 Listen All=7

SetBaudrate

The bit rate of a CAN channel can be selected one of two ways. It can either be selected from a list of common bit rates (SetBaudrate=1) or the user can specify the CAN timing parameters (SetBaudrate=0)

TqProp

Propagation delay

TqSeg1

Phase 1 segment

TqSeg2

Phase 2 segment

TqSync

Syncro jump width

auto_baud

Enables the auto bitrate feature. 1 = enable, 0 = disable.

innerFrameDelay25us

transceiver_mode

Currently Not used.

class ics.CmISO157652RxMessage

Bases: object

CmISO157652RxMessage object

blockSize

Overrides the block size that the receiver reports, see overrideBlockSize. Set to J2534's BS_TX if <= 0xFF

cf_timeout

max timeout (ms) for waiting on consecutive frame. Set this to N_CR_MAX's value in J2534

extendedAddress

Extended Address byte of transmitter. see ext_address_enable, not supported

fc_id

flow control arbid to transmit in flow control (from neoVI to ECU)

flags

flowControlExtendedAddress

Expected Extended Address byte of response from receiver. see fc_ext_address_enable, not supported

id

arbid of transmitted frames (CAN id to transmit to)

id_mask

ArbId filter mask for frames from transmitter (from ECU to neoVI)

padding

The padding byte to use to fill the unused portion of * transmitted CAN frames (flow control), see paddingEnable.

reserved**stMin**

Minimum separation time (between consecutive frames) to report in flow control response

vs_netid

The netid of the message (determines which network to decode receives), not supported

class ics.CmISO157652TxMessage

Bases: object

CmISO157652TxMessage object

blockSize

Overrides the block size that the receiver reports, see overrideBlockSize. Set to J2534's BS_TX if <= 0xFF

data

The data

extendedAddress

Extended Address byte of transmitter. see ext_address_enable, not supported

fc_id

flow control arb id filter value (response id from receiver)

fc_id_mask

The flow control arb filter mask (response id from receiver)

flags**flowControlExtendedAddress**

Expected Extended Address byte of response from receiver. see fc_ext_address_enable, not supported

fs_timeout

max timeout (ms) for waiting on flow control responses. Set this to N_BS_MAX's value if J2534

fs_wait

max timeout (ms) for waiting on flow control response after receiving flow control * with flow status set to WAIT. Set this to N_BS_MAX's value if J2534.

id

arbId of transmitted frames (CAN id to transmit to)

num_bytes

Number of data bytes

padding

The padding byte to use to fill the unused portion of * transmitted CAN frames (single frame, first frame, consecutive frame) *

stMin

Overrides the stMin that the receiver reports, see overrideSTmin. Set to J2534's STMIN_TX if <= 0xFF

tx_index**vs_netid**

The netid of the message (determines which network to transmit on), not supported

```
class ics.CyanSettings
    Bases: object
    CyanSettings object
    ain_sample_period
    ain_threshold
    can1
        ics.CanSettings Object
    can2
        ics.CanSettings Object
    can3
        ics.CanSettings Object
    can4
        ics.CanSettings Object
    can5
        ics.CanSettings Object
    can6
        ics.CanSettings Object
    can7
        ics.CanSettings Object
    can8
        ics.CanSettings Object
    can_switch_mode
    canfd1
        ics.CanFdSettings Object
    canfd2
        ics.CanFdSettings Object
    canfd3
        ics.CanFdSettings Object
    canfd4
        ics.CanFdSettings Object
    canfd5
        ics.CanFdSettings Object
    canfd6
        ics.CanFdSettings Object
    canfd7
        ics.CanFdSettings Object
    canfd8
        ics.CanFdSettings Object
    digitalIoThresholdEnable
    digitalIoThresholdTicks
    disableUsbCheckOnBoot
    enableLatencyTest
```

ethernet
ics.EthernetSettings Object

idle_wakeup_network_enables_3

iso15765_separation_time_offset

iso9141_kwp_settings_1
Iso9141Keyword2000Settings Object

iso9141_kwp_settings_2
Iso9141Keyword2000Settings Object

iso9141_kwp_settings_3
Iso9141Keyword2000Settings Object

iso9141_kwp_settings_4
Iso9141Keyword2000Settings Object

iso_msg_termination_1
0 - use inner frame time, 1 - GME CIM-SCL

iso_msg_termination_2
0 - use inner frame time, 1 - GME CIM-SCL

iso_msg_termination_3
0 - use inner frame time, 1 - GME CIM-SCL

iso_msg_termination_4
0 - use inner frame time, 1 - GME CIM-SCL

iso_parity_1
0 - no parity, 1 - event, 2 - odd

iso_parity_2
0 - no parity, 1 - event, 2 - odd

iso_parity_3
0 - no parity, 1 - event, 2 - odd

iso_parity_4
0 - no parity, 1 - event, 2 - odd

lin1
ics.LinSettings Object

lin2
ics.LinSettings Object

lin3
ics.LinSettings Object

lin4
ics.LinSettings Object

lin5
ics.LinSettings Object

lin6
ics.LinSettings Object

lsft1
ics.CanSettings Object

lsft2
ics.CanSettings Object

misc_io_analog_enable

misc_io_initial_ddr

misc_io_initial_latch

misc_io_on_report_events

misc_io_report_period

network_enabled_on_boot

network_enables

network_enables_2

network_enables_3

perf_en

pwr_man_enable

pwr_man_timeout

reserved

slaveVnetA

slaveVnetB

swcan1
ics.SWCanSettings Object

swcan2
ics.SWCanSettings Object

termination_enables

text_api
ics.TextApiSettings Object

class ics.DeviceSettings
Bases: object

DeviceSettings object

DeviceSettingType

cyan
ics.CyanSettings Object

fire
ics.FireSettings Object

radgalaxy
ics.RadGalaxySettings Object

radstar2
ics.RadStar2Settings Object

vcan3
ics.Vcan3Settings Object

vcan4
ics.Vcan4Settings Object

vcan4_12

ics.Vcan412Settings Object

vividcan

ics.VividCANSettings Object

class ics.EthernetSettings

Bases: object

EthernetSettings object

auto_neg

duplex

led_mode

link_speed

rsvd

class ics.Fire2DeviceStatus

Bases: object

Fire2DeviceStatus object

backupPowerEnabled

backupPowerGood

ethernetActivationLineEnabled

usbHostPowerEnabled

class ics.FireSettings

Bases: object

FireSettings object

ain_sample_period

ain_threshold

can1

ics.CanSettings Object

can2

ics.CanSettings Object

can3

ics.CanSettings Object

can4

ics.CanSettings Object

cgi_baud

cgi_chksum_enable

cgi_enable_reserved

cgi_rx_ifs_bit_times

cgi_tx_ifs_bit_times

fast_init_network_enables_1

fast_init_network_enables_2

iso15765_separation_time_offset
iso9141_kwp_enable_reserved
iso9141_kwp_settings
Iso9141Keyword2000Settings Object
iso9141_kwp_settings_2
Iso9141Keyword2000Settings Object
iso9141_kwp_settings_3
Iso9141Keyword2000Settings Object
iso9141_kwp_settings_4
Iso9141Keyword2000Settings Object
iso_msg_termination
0 - use inner frame time, 1 - GME CIM-SCL
iso_msg_termination_2
0 - use inner frame time, 1 - GME CIM-SCL
iso_msg_termination_3
0 - use inner frame time, 1 - GME CIM-SCL
iso_msg_termination_4
0 - use inner frame time, 1 - GME CIM-SCL
iso_parity
0 - no parity, 1 - event, 2 - odd
iso_parity_2
0 - no parity, 1 - event, 2 - odd
iso_parity_3
0 - no parity, 1 - event, 2 - odd
iso_parity_4
0 - no parity, 1 - event, 2 - odd
iso_tester_pullup_enable
lin1
ics.LinSettings Object
lin2
ics.LinSettings Object
lin3
ics.LinSettings Object
lin4
ics.LinSettings Object
lsft
ics.CanSettings Object
misc_io_analog_enable
misc_io_initial_ddr
misc_io_initial_latch
misc_io_on_report_events
misc_io_report_period

```
network_enabled_on_boot
network_enables
network_enables_2
perf_en
pwm_man_timeout
pwr_man_enable
swcan
    ics.SWCanSettings Object
text_api
    ics.TextApiSettings Object
uart
    ics.UartSettings Object
uart2
    ics.UartSettings Object
vnetBits
class ics.IcsDeviceStatus
    Bases: object
    IcsDeviceStatus object
    fire2Status
        ics.Fire2DeviceStatus Object
    vcan4Status
        ics.Vcan4DeviceStatus Object
class ics.Iso9141Keyword2000InitSteps
    Bases: object
    Iso9141Keyword2000InitSteps object
    k
    l
    time_500us
class ics.Iso9141Keyword2000Settings
    Bases: object
    Iso9141Keyword2000Settings object
    Baudrate
    brgh
    checksum_enabled
    init_steps
        Tuple of Iso9141Keyword2000InitSteps
    p2_500us
    p3_500us
    p4_500us
```


spbrg

class ics.LinSettings

Bases: object

LinSettings object

Baudrate

MasterResistor

Mode

brgh

spbrg

class ics.NeoDevice

Bases: object

NeoDevice object

AutoHandleClose

When NeoDevice is freed the handle will automatically be closed, if true.

DeviceType

Handle

IsOpen

This contains the handle returned from icsneoOpenDevice() API. If uncertain, don't use this.

MaxAllowedClients

Name

String describing DeviceType, extension to Python api only.

NumberOfClients

SerialNumber

class ics.OpEthGeneralSettings

Bases: object

OpEthGeneralSettings object

bEnReportLinkQuality

bTapEnPtp

bTapEnSwitch

reserved0

tapPair0

tapPair1

tapPair2

tapPair3

tapPair4

tapPair5

ucInterfaceType

class ics.OpEthSettings

Bases: object

OpEthSettings object

preemption_en

reserved0

ucConfigMode

class ics.RadGalaxySettings

Bases: object

RadGalaxySettings object

ain_sample_period

ain_threshold

can1

ics.CanSettings Object

can2

ics.CanSettings Object

can3

ics.CanSettings Object

can4

ics.CanSettings Object

can5

ics.CanSettings Object

can6

ics.CanSettings Object

can7

ics.CanSettings Object

can8

ics.CanSettings Object

can_switch_mode

canfd1

ics.CanFdSettings Object

canfd2

ics.CanFdSettings Object

canfd3

ics.CanFdSettings Object

canfd4

ics.CanFdSettings Object

canfd5

ics.CanFdSettings Object

canfd6

ics.CanFdSettings Object

canfd7

ics.CanFdSettings Object

canfd8
ics.CanFdSettings Object

idle_wakeup_network_enables_1

idle_wakeup_network_enables_2

idle_wakeup_network_enables_3

iso15765_separation_time_offset

iso9141_kwp_settings_1
Iso9141Keyword2000Settings Object

iso_msg_termination_1
0 - use inner frame time, 1 - GME CIM-SCL

iso_parity_1
0 - no parity, 1 - event, 2 - odd

lin1
ics.LinSettings Object

misc_io_analog_enable

misc_io_initial_ddr

misc_io_initial_latch

misc_io_on_report_events

misc_io_report_period

network_enabled_on_boot

network_enables

network_enables_2

network_enables_3

opEth1
ics.OpEthSettings Object

opEth10
ics.OpEthSettings Object

opEth11
ics.OpEthSettings Object

opEth12
ics.OpEthSettings Object

opEth2
ics.OpEthSettings Object

opEth3
ics.OpEthSettings Object

opEth4
ics.OpEthSettings Object

opEth5
ics.OpEthSettings Object

opEth6
ics.OpEthSettings Object

opEth7
ics.OpEthSettings Object

opEth8
ics.OpEthSettings Object

opEth9
ics.OpEthSettings Object

opEthGen
ics.OpEthGeneralSettings Object

perf_en

pwr_man_enable

pwr_man_timeout

swcan1
ics.SWCanSettings Object

swcan2
ics.SWCanSettings Object

text_api
ics.TextApiSettings Object

class ics.RadStar2Settings
Bases: object

RadStar2Settings object

ain_sample_period

ain_threshold

can1
ics.CanSettings Object

can2
ics.CanSettings Object

can_switch_mode

canfd1
ics.CanFdSettings Object

canfd2
ics.CanFdSettings Object

hwComLatencyTestEn

idle_wakeup_network_enables_1

idle_wakeup_network_enables_2

idle_wakeup_network_enables_3

iso15765_separation_time_offset

iso9141_kwp_settings_1
Iso9141Keyword2000Settings Object

iso_9141_kwp_enable_reserved

iso_msg_termination_1
0 - use inner frame time, 1 - GME CIM-SCL

iso_parity_1
0 - no parity, 1 - event, 2 - odd

lin1
ics.LinSettings Object

misc_io_analog_enable

misc_io_initial_ddr

misc_io_initial_latch

misc_io_on_report_events

misc_io_report_period

network_enabled_on_boot

network_enables

network_enables_2

network_enables_3

opEth1
ics.OpEthSettings Object

opEth2
ics.OpEthSettings Object

opEthGen
ics.OpEthGeneralSettings Object

pc_com_mode

perf_en

pwr_man_enable

pwr_man_timeout

text_api
ics.TextApiSettings Object

timeSyncSettings
ics.TimesyncSettings Object

class ics.SWCanSettings

Bases: object

SWCanSettings object

BRP

Baudrate

The bit rate of a CAN channel can be selected from a list of common bit rates Write the correct enumeration for the desired bit rate and ensure that SetBaudrate is 1(auto)

Mode

CAN controller mode when the neoVI device goes online or runs a CoreMini script. Normal=0 Disabled=1 Listen Only=3 Listen All=7

RESERVED

SetBaudrate

The bit rate of a CAN channel can be selected one of two ways. It can either be selected from a list of common bit rates (SetBaudrate=1) or the user can specify the CAN timing parameters (SetBaudrate=0)

TqProp

Propagation delay

TqSeg1

Phase 1 segment

TqSeg2

Phase 2 segment

TqSync

Syncro jump width

auto_baud

Enables the auto bitrate feature. 1 = enable, 0 = disable.

high_speed_auto_switch**transceiver_mode**

Currently Not used.

class ics.SpyMessage

Bases: object

SpyMessage object

AckBytes**ArbIDOrHeader****Data****DescriptionID**

Not Used

ExtraDataPtr**ExtraDataPtrEnabled****MessagePieceID**

Not Used

MiscData**NetworkID**

This value is used to identify which network this message was received on.

NetworkID2

This value is used to identify which network this message was received on.

NodeID

Not Used

NumberBytesData

Holds the number of bytes in the Data(1 to 8) array or the number of bytes in a CAN remote frame (The DLC).

NumberBytesHeader

Used for J1850/ISO messages. It indicates how many bytes are stored in the Header(1 to 4) array.

Protocol

Valid values are SPY_PROTOCOL_CAN, SPY_PROTOCOL_J1850VPW, and SPY_PROTOCOL_ISO9141.

StatusBitField**StatusBitField2**

StatusBitField3

StatusBitField4

TimeHardware

Hardware time stamp. The TimeStamp is reset on device open

TimeHardware2

Hardware time stamp. The TimeStamp is reset on device open

TimeStampHardwareID

This is an identifier of what type of hardware timestamp is used. Since neoVI's timestamp is always the same, this doesn't change.

TimeStampSystemID

This is an identifier of what type of system timestamp is used. Since WIN32 neoVI's timestamp is always the same, from the timeGetTime API, this doesn't change.

TimeSystem

TimeSystem is loaded with the value received from the timeGetTime call in the WIN32 multimedia API.

TimeSystem2

TimeSystem is loaded with the value received from the timeGetTime call in the WIN32 multimedia API.

noExtraDataPtrCleanup

Tells Python to not clean up ExtraDataPtrMemory, If this is enabled. Ignore, if unsure.

class ics.SpyMessageJ1850

Bases: object

SpyMessageJ1850 object

AckBytes

Data

DescriptionID

Not Used

ExtraDataPtr

ExtraDataPtrEnabled

Header

MessagePieceID

Not Used

MiscData

NetworkID

This value is used to identify which network this message was received on.

NetworkID2

This value is used to identify which network this message was received on.

NodeID

Not Used

NumberBytesData

Holds the number of bytes in the Data(1 to 8) array or the number of bytes in a CAN remote frame (The DLC).

NumberBytesHeader

Used for J1850/ISO messages. It indicates how many bytes are stored in the Header(1 to 4) array.

Protocol

Valid values are SPY_PROTOCOL_CAN, SPY_PROTOCOL_J1850VPW, and SPY_PROTOCOL_ISO9141.

StatusBitField**StatusBitField2****StatusBitField3****StatusBitField4****TimeHardware**

Hardware time stamp. The TimeStamp is reset on device open

TimeHardware2

Hardware time stamp. The TimeStamp is reset on device open

TimeStampHardwareID

This is an identifier of what type of hardware timestamp is used. Since neoVI's timestamp is always the same, this doesn't change.

TimeStampSystemID

This is an identifier of what type of system timestamp is used. Since WIN32 neoVI's timestamp is always the same, from the timeGetTime API, this doesn't change.

TimeSystem

TimeSystem is loaded with the value received from the timeGetTime call in the WIN32 multimedia API.

TimeSystem2

TimeSystem is loaded with the value received from the timeGetTime call in the WIN32 multimedia API.

noExtraDataPtrCleanup

Tells Python to not clean up ExtraDataPtrMemory, If this is enabled. Ignore, if unsure.

class ics.TextApiSettings

Bases: object

TextApiSettings object

can1_options

Sets the length of the Arbitration ID's. Set to 1 for Extended and 0 for Standard

can1_rx_id

Sets or Reads the Arbitration ID for Sending Receiving API commands

can1_tx_id

Sets or Reads the Arbitration ID for Sending Text API commands

can2_options**can2_rx_id****can2_tx_id****can3_options****can3_rx_id****can3_tx_id****can4_options****can4_rx_id****can4_tx_id**

network_enables

Bitfield telling which network to support Text API.

class ics.TimesyncSettings

Bases: object

TimesyncSettings object

MasterEnable**MasterNetwork****SlaveEnable****SlaveNetwork****class ics.UartSettings**

Bases: object

UartSettings object

Baudrate

Holds the baud rate for the UART Connection. An example value could be 10417 or 9600

bOptions

Bitfield containing UART Options Invert TX=1, Invert RX=2, Half Duplex=4

brgh**flow_control**

Set to 0 for no flow control and 1 for simple CTS RTS

parity

Sets the Parity type. Valid values are None=0, Even=1, Odd=2

reserved_1**spbrg****stop_bits**

Sets the number of stop bits to use. Valid values are One=1, Two=2

class ics.Vcan3Settings

Bases: object

Vcan3Settings object

can1

ics.CanSettings Object

can2

ics.CanSettings Object

isol5765_separation_time_offset**misc_io_initial_ddr****misc_io_initial_latch****misc_io_on_report_events****misc_io_report_period****network_enabled_on_boot****network_enables****perf_en**

```
class ics.Vcan412Settings
    Bases: object
    Vcan412Settings object

    can1
        ics.CanSettings Object

    can2
        ics.CanSettings Object

    canfd1
        ics.CanFdSettings Object

    canfd2
        ics.CanFdSettings Object

    disableUsbCheckOnBoot
        flags

    enableLatencyTest
        flags

    iso15765_separation_time_offset

    network_enabled_on_boot

    network_enables

    perf_en

    pwr_man_enable

    pwr_man_timeout

    reserved
        flags

    termination_enables

    text_api
        ics.TextApiSettings Object

class ics.Vcan4DeviceStatus
    Bases: object
    Vcan4DeviceStatus object

    ethernetActivationLineEnabled

class ics.Vcan4Settings
    Bases: object
    Vcan4Settings object

    can1
        ics.CanSettings Object

    can2
        ics.CanSettings Object

    can3
        ics.CanSettings Object

    can4
        ics.CanSettings Object
```

canfd1
ics.CanFdSettings Object

canfd2
ics.CanFdSettings Object

canfd3
ics.CanFdSettings Object

canfd4
ics.CanFdSettings Object

enableLatencyTest
flags

enablePcEthernetComm
flags

ethernet
ics.EthernetSettings Object

iso15765_separation_time_offset

iso9141_kwp_settings_1
Iso9141Keyword2000Settings Object

iso_9141_kwp_enable_reserved

iso_msg_termination_1

iso_parity_1

lin1
ics.LinSettings Object

network_enabled_on_boot

network_enables

network_enables_2

network_enables_3

perf_en

pwr_man_enable

pwr_man_timeout

reserved
flags

termination_enables

text_api
ics.TextApiSettings Object

class ics.VcanRFSettings

Bases: object

VcanRFSettings object

can1
ics.CanSettings Object

can2
ics.CanSettings Object

can3
ics.CanSettings Object

can4
ics.CanSettings Object

idle_wakeup_network_enables_1

idle_wakeup_network_enables_2

iso15765_separation_time_offset

iso9141_kwp_enable_reserved

iso9141_kwp_settings
ics.Iso9141Keyword2000Settings Object

iso9141_kwp_settings_2
ics.Iso9141Keyword2000Settings Object

iso_msg_termination
0 - use inner frame time, 1 - GME CIM-SCL

iso_msg_termination_2
0 - use inner frame time, 1 - GME CIM-SCL

iso_parity
0 - no parity, 1 - event, 2 - odd

iso_parity_2
0 - no parity, 1 - event, 2 - odd

iso_tester_pullup_enable

lin1
ics.LinSettings Object

lin2
ics.LinSettings Object

misc_io_analog_enable

misc_io_initial_ddr

misc_io_initial_latch

misc_io_on_report_events

misc_io_report_period

network_enabled_on_boot

network_enables

network_enables_2

perf_en

pwr_man_enable
0 - off, 1 - sleep enabled, 2- idle enabled (fast wakeup)

pwr_man_timeout

class ics.VividCANSettings
Bases: object
VividCANSettings object

can1
ics.CanSettings Object

can_switch_mode

disableUsbCheckOnBoot
flags

ecu_id

enableLatencyTest
flags

iso15765_separation_time_offset

lsftcan1
ics.CanSettings Object

network_enabled_on_boot

network_enables

perf_en

pwr_man_enable

pwr_man_timeout

reserved
flags

swcan1
ics.CanSettings Object

termination_enables

ics.ClosePort()

Note: Compatibility Function Identical to PEP8 compliant *ics.close_device()* method.

ics.EnableBusVoltageMonitor()

Note: Compatibility Function Identical to PEP8 compliant *ics.enable_bus_voltage_monitor()* method.

ics.EnableNetworkCom()

Note: Compatibility Function Identical to PEP8 compliant *ics.enable_network_com()* method.

ics.FindNeoDevices()

Note: Compatibility Function Identical to PEP8 compliant *ics.find_devices()* method.

`ics.FirmwareUpdateRequired()`

Note: Compatibility Function Identical to PEP8 compliant `ics.firmware_update_required()` method.

`ics.ForceFirmwareUpdate()`

Note: Compatibility Function Identical to PEP8 compliant `ics.force_firmware_update()` method.

`ics.GetActiveVNETChannel()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_active_vnet_channel()` method.

`ics.GetBackupPowerEnabled()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_backup_power_enabled()` method.

`ics.GetBackupPowerReady()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_backup_power_ready()` method.

`ics.GetBusVoltage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_bus_voltage()` method.

`ics.GetDLLFirmwareInfo()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_dll_firmware_info()` method.

`ics.GetDLLVersion()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_dll_version()` method.

`ics.GetDeviceStatus()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_device_status()` method.

`ics.GetErrorMessages()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_error_messages()` method.

`ics.GetFireSettings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_device_settings()` method.

`ics.GetHWFirmwareInfo()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_hw_firmware_info()` method.

`ics.GetLastAPIError()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_last_api_error()` method.

`ics.GetMessages()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_messages()` method.

`ics.GetPerformanceParameters()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_performance_parameters()` method.

`ics.GetRTC()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_rtc()` method.

`ics.GetSerialNumber()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_serial_number()` method.

`ics.GetTimeStampForMsg()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_timestamp_for_msg()` method.

`ics.GetVCAN3Settings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_device_settings()` method.

`ics.ISO15765_DisableNetworks()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_disable_networks()` method.

`ics.ISO15765_EnableNetworks()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_enable_networks()` method.

`ics.ISO15765_ReceiveMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_receive_message()` method.

`ics.ISO15765_TransmitMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_transmit_message()` method.

`ics.LoadDefaultSettings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.load_default_settings()` method.

`ics.OpenNeoDevice()`

Note: Compatibility Function Identical to PEP8 compliant `ics.open_device()` method.

`ics.ReadSDCard()`

Note: Compatibility Function Identical to PEP8 compliant `ics.read_sdcard()` method.

`ics.RequestEnterSleepMode()`

Note: Compatibility Function Identical to PEP8 compliant `ics.request_enter_sleep_mode()` method.

`ics.ScriptClear()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_clear()` method.

`ics.ScriptGetFBlockStatus()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_get_fblock_status()` method.

`ics.ScriptGetScriptStatus()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_get_status()` method.

`ics.ScriptGetScriptStatusEx()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_script_status()` method.

`ics.ScriptLoad()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_load()` method.

`ics.ScriptReadAppSignal()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_read_app_signal()` method.

`ics.ScriptReadRxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_read_rx_message()`

method.

`ics.ScriptReadTxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_read_tx_message()` method.

`ics.ScriptStart()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_start()` method.

`ics.ScriptStartFBlock()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_start_fblock()` method.

`ics.ScriptStop()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_stop()` method.

`ics.ScriptStopFBlock()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_stop_fblock()` method.

`ics.ScriptWriteAppSignal()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_write_app_signal()` method.

`ics.ScriptWriteRxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_write_rx_message()` method.

`ics.ScriptWriteTxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_write_tx_message()` method.

`ics.SetActiveVNETChannel()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_active_vnet_channel()` method.

`ics.SetBackupPowerEnabled()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_backup_power_enabled()` method.

`ics.SetBitRate()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_bit_rate()` method.

`ics.SetBitRateEx()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_bit_rate_ex()` method.

`ics.SetContext()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_context()` method.

`ics.SetFDBitRate()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_fd_bit_rate()` method.

`ics.SetFireSettings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_device_settings()` method.

`ics.SetRTC()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_rtc()` method.

`ics.SetReflashDisplayCallback()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_reflash_callback()` method.

`ics.SetVCAN3Settings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_device_settings()` method.

`ics.TxMessages()`

Note: Compatibility Function Identical to PEP8 compliant `ics.transmit_messages()` method.

`ics.ValidateHObject()`

Note: Compatibility Function Identical to PEP8 compliant `ics.validate_hobject()` method.

`ics.WriteSDCard()`

Note: Compatibility Function Identical to PEP8 compliant `ics.write_sdcard()` method.

`ics.base36enc(serial)`

Converts a decimal serial number to base36.

Args: serial (int): serial number.

Raises: `ics.RuntimeError`

Returns: Str: Serial Number

```
>>> ics.base36enc(device.SerialNumber)
CY0024
```

`ics.close_device(device)`

Closes the device.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Error Count (int)

```
>>> for device in ics.find_devices():
...     ics.open_device(device)
...     # Do something with the device...
...     ics.close_device(device)
... 
```

Note: `ics.NeoDevice` will automatically close the device when it goes out of scope.

ics.coremini_clear (*device*, *location*)

Clears the CoreMini into the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

location (int): Accepts *ics.SCRIPT_LOCATION_FLASH_MEM*, *ics.SCRIPT_LOCATION_SDCARD*, or *ics.SCRIPT_LOCATION_VCAN3_MEM*

Raises: *ics.RuntimeError*

Returns: None.

```
>>> device = ics.open_device()
>>> ics.coremini_clear(device, ics.SCRIPT_LOCATION_SDCARD)
```

ics.coremini_get_fblock_status (*device*, *index*)

Gets the status of a Coremini Function Block at *index* on *device*.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

index (int): Index of the function block.

Raises: *ics.RuntimeError*

Returns: None on Success.

```
>>> device = ics.open_device()
>>> ics.coremini_get_fblock_status(device, 1)
True
```

ics.coremini_get_status (*device*)

Gets the status of the CoreMini in the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: True if running, otherwise False.

```
>>> device = ics.open_device()
>>> ics.coremini_get_status(device)
>>>
```

ics.coremini_load (*device*, *coremini*, *location*)

Loads the CoreMini into the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

coremini (str/tuple): Use string to load from file, Use Tuple if file data.

location (int): Accepts *ics.SCRIPT_LOCATION_FLASH_MEM*, *ics.SCRIPT_LOCATION_SDCARD*, or *ics.SCRIPT_LOCATION_VCAN3_MEM*

Raises: *ics.RuntimeError*

Returns: None.

```
>>> device = ics.open_device()
>>> ics.coremini_load(device, 'cmvspy.vs3cmb', ics.SCRIPT_LOCATION_SDCARD)
```

ics.coremini_read_app_signal (*device*, *index*)

Gets the value of a Coremini application signal at *index* on *device*.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

index (int): Index of the application signal.

Raises: *ics.RuntimeError*

Returns: int on Success.

```
>>> device = ics.open_device()
>>> ics.coremini_read_app_signal(device, 1)
52
```

ics.coremini_read_rx_message (*device, index, j1850=False*)

Gets the value of a Coremini Message at *index* on *device*.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

index (int): Index of the application signal.

j1850 (bool): Use *ics.SpyMessageJ1850* instead.

Raises: *ics.RuntimeError*

Returns: *ics.SpyMessage* Success.

```
>>> device = ics.open_device()
>>> msg = ics.coremini_read_tx_message(device, 0)
```

ics.coremini_read_tx_message (*device, index, j1850=False*)

Gets the value of a Coremini Message at *index* on *device*.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

index (int): Index of the application signal.

j1850 (bool): Use *ics.SpyMessageJ1850* instead.

Raises: *ics.RuntimeError*

Returns: *ics.SpyMessage* Success.

```
>>> device = ics.open_device()
>>> msg = ics.coremini_read_tx_message(device, 0)
```

ics.coremini_start (*device, location*)

Starts the CoreMini into the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

location (int): Accepts *ics.SCRIPT_LOCATION_FLASH_MEM*, *ics.SCRIPT_LOCATION_SDCARD*, or *ics.SCRIPT_LOCATION_VCAN3_MEM*

Raises: *ics.RuntimeError*

Returns: None.

```
>>> device = ics.open_device()
>>> ics.coremini_start(device, ics.SCRIPT_LOCATION_SDCARD)
```

ics.coremini_start_fblock (*device, index*)

Starts a Coremini Function Block at *index* on *device*.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

index (int): Index of the function block.

Raises: `ics.RuntimeError`

Returns: None on Success.

```
>>> device = ics.open_device()
>>> ics.coremini_start_fblock(device, 1)
```

`ics.coremini_stop(device)`

Stops the CoreMini into the device.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: None.

```
>>> device = ics.open_device()
>>> ics.coremini_stop(device)
```

`ics.coremini_stop_fblock(device, index)`

Stops a Coremini Function Block at *index* on *device*.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

index (int): Index of the function block.

Raises: `ics.RuntimeError`

Returns: None on Success.

```
>>> device = ics.open_device()
>>> ics.coremini_stop_fblock(device, 1)
```

`ics.coremini_write_app_signal(device, index, value)`

Sets the value of a Coremini application signal at *index* on *device*.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

index (int): Index of the application signal.

value (int): New value of the application signal.

Raises: `ics.RuntimeError`

Returns: None on Success.

```
>>> device = ics.open_device()
>>> ics.coremini_write_app_signal(device, 1, 52)
>>>
```

`ics.coremini_write_rx_message(device, index, TODO)`

TODO

`ics.coremini_write_tx_message(device, index, msg)`

TODO

`ics.create_neovi_radio_message(Relay1, Relay2, Relay3, Relay4, Relay5, LED6, LED5, MSB_report_rate, LSB_report_rate, analog_change_report_rate, relay_timeout)`

Python API only. Generates data bytes for use with neoVI RADI/O CAN Messages

Kwargs: Relay1 (boolean): Enable/Disable Relay1

Relay2 (boolean): Enable/Disable Relay2

Relay3 (boolean): Enable/Disable Relay3

Relay4 (boolean): Enable/Disable Relay4

Relay5 (boolean): Enable/Disable Relay5

LED5 (boolean): Enable/Disable LED5

LED6 (boolean): Enable/Disable LED6

MSB_report_rate (int): MSB Report Rate in ms (0-255)

LSB_report_rate (int): LSB Report Rate in ms (0-255)

analog_change_report_rate (int): Analog change report rate

relay_timeout (int): Relay Timeout (0-255)*255ms

Returns:

Tuple of data bytes for use with `ics.SpyMessage`

Raises: `ics.RuntimeError`

```
>>> msg = ics.SpyMessage()
>>> msg.Data = ics.create_neovi_radio_message(Relay1=True, Relay4=False,
↳ LED6=True, MSB_report_rate=10)
>>> msg.Data
(65, 10, 0, 0, 0)
```

`ics.enable_bus_voltage_monitor` (*device, enable, reserved*)

Enable or disable bus voltage monitoring.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

enable (bool): int

reserved (int): int: Optional. Should be set to zero. Don't set, if unsure.

Raises: `ics.RuntimeError`

Returns: None.

```
>>> import ics
>>> d = ics.open_device()
>>> status = ics.enable_bus_voltage_monitor(d, 1)
>>>
```

`ics.enable_network_com` (*device, enable, net_id*)

Enable or disable network communication.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

enable (bool): bool

net_id (int): int: Optional. If left blank, disables/enables all networks.

Raises: `ics.RuntimeError`

Returns: None.

```
>>> import ics
>>> d = ics.open_device()
>>> status = ics.enable_network_com(d, True)
>>>
```


`ics.find_devices(device_type=ics.NEODEVICE_ALL)`

Finds all connected devices and returns a tuple of `ics.NeoDevice` for use in `ics.open_device()`

Args: `device_type` (int): Accepts `ics.NEODEVICE_*` Macros

New in 3.0 (803):

`device_type` (List/Tuple): Accepts a Container of `ics.NEODEVICE_*` Macros

`stOptionsOpenNeoEx` (int): Usually `ics.NETID_CAN`, if needed

Raises: `ics.RuntimeError`

Returns: Tuple of `ics.NeoDevice` for use in `ics.open_device()`

```
>>> for device in ics.find_devices():
...     print(device.Name, device.SerialNumber)
...
neoVI FIRE 59886
```

New in 3.0 (803):

```
>>> for device in ics.find_devices([ics.NEODEVICE_FIRE, ics.NEODEVICE_VCAN3]):
...     print(device.Name, device.SerialNumber)
...
neoVI FIRE 59886
```

`ics.firmware_update_required(device)`

Determines if the device firmware needs flashing.

Args: `device` (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

```
>>> ics.force_firmware_update(device)
True
```

`ics.force_firmware_update(device)`

Forces the device to flash firmware.

Args: `device` (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

```
>>> ics.force_firmware_update(device)
True
```

`ics.get_active_vnet_channel(device)`

Gets active vnet channel for the device.

Args: `device` (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Int: Returns active vnet channel.

`ics.get_backup_power_enabled(device)`

Returns the device backup power enabled for the device.

Args: `device` (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

`ics.get_backup_power_ready(device)`

Returns the device backup power is ready for the device.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

`ics.get_bus_voltage(device, reserved)`

Reads bus voltage. (`ics.enable_bus_voltage_monitor`) needs to be called first.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

reserved (int): int: Optional. Should be set to zero. Don't set, if unsure.

Raises: `ics.RuntimeError`

Returns: Int: value returned is in mV.

```
>>> import ics
>>> d = ics.open_device()
>>> status = ics.enable_bus_voltage_monitor(d, 1)
>>> ics.get_bus_voltage(d)
12000
>>>
```

`ics.get_device_settings(device, vnet_slot)`

Gets the settings in the device. vnet_slot defaults to `ics.PlasmaIonVnetChannelMain`

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: `ics.DeviceSettings`

```
>>> d = ics.open_device()
>>> d.Name
'neoVI ION'
>>> d.SerialNumber
404444
>>> s = ics.get_device_settings(d)
>>> s.DeviceSettingType
2
>>> s.cyan
<ics.CyanSettings object at 0x01E61B40>
>>> s.cyan.canfd1.FDMode
4
>>> s2.cyan
<ics.CyanSettings object at 0x02B113C8>
>>> s2 = ics.get_device_settings(d, ics.PlasmaIonVnetChannelA)
>>> s2.DeviceSettingType
2
>>> s2.cyan.canfd1.FDMode
4
```

`ics.get_device_status(device)`

Returns the device status.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: (*ics.IcsDeviceStatus*).

```
>>> import ics
>>> d = ics.open_device()
>>> status = ics.get_device_status(d)
>>> status.fire2Status.ethernetActivationLineEnabled
0
```

ics.get_dll_firmware_info (device)

Returns the DLL firmware info for the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: (*ics.ApiFirmwareInfo*)

```
>>> device = ics.open_device()
>>> info = ics.get_dll_firmware_info(device)
>>> info.iAppMajor
7
>>> info.iAppMinor
55
>>>
```

ics.get_dll_version (device)

Gets the DLL version.

Args: None

Raises: *ics.RuntimeError*

Returns: Int: DLL Version

```
>>> ics.get_dll_version()
700
```

ics.get_error_messages (device[, j1850, timeout])

Gets the error message(s) on the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: list of tuple`s. :class:`tuple` contents: (error_number, description_short, description_long, severity, restart_needed)

```
>>> device = ics.open_device()
>>> errors = ics.get_error_messages(device)
```

ics.get_hw_firmware_info (device)

Returns the device firmware info for the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: (*ics.ApiFirmwareInfo*)

```
>>> device = ics.open_device()
>>> info = ics.get_hw_firmware_info(device)
>>> info.iAppMajor
7
>>> info.iAppMinor
55
>>>
```

ics.get_last_api_error(*device*)

Gets the error message from the last API call.

Args: *device* (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: Tuple: (error, description short, description long, severity, restart needed)

```
>>> device = ics.open_device()
>>> try:
...     msg = ics.coremini_read_tx_message(device, 0)
... except ics.RuntimeError as ex:
...     print(ex)
...     print(ics.get_last_api_error(device))
...
Error: coremini_read_tx_message(): icsneoScriptReadTxMessage() Failed
(224, 'Invalid Message Index for script.', 'Invalid Message Index for script.
↪', 16, 0)
```

ics.get_library_path()

ics.get_messages(*device*[, *j1850*, *timeout*])

Gets the message(s) on the device.

Args: *device* (*ics.NeoDevice*): *ics.NeoDevice*

j1850 (bool): Return *ics.SpyMessageJ1850* instead.

imeout (float): Optional timeout to wait for messages in seconds (0.1 = 100ms).

Raises: *ics.RuntimeError*

Returns: tuple of two items. First item is a tuple of *ics.SpyMessage* and second is the error count.

```
>>> device = ics.open_device()
>>> messages, errors = ics.get_messages(device)
>>> len(messages)
14
>>> hex(messages[0].ArbIDOrHeader)
'0x160'
>>> messages[0].Data
(36, 11, 11, 177, 37, 3, 11, 199)
>>> errors
0
```

ics.get_performance_parameters(*device*)

Gets the Performance Parameters on *device*.

Args: *device* (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: Tuple on Success: (buffer count, buffer max, overflow count, reserved, reserved, reserved, reserved, reserved)

```
>>> device = ics.open_device()
>>> ics.get_performance_parameters(device)
(0, 24576, 0, 0, 0, 0, 0, 0)
```

`ics.get_rtc(device)`

Gets the Real-Time Clock of the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: Tuple: (datetime.datetime object, offset in seconds)

```
>>> device = ics.open_device()
>>> ics.get_rtc(device)
(datetime.datetime(2014, 9, 10, 17, 45, 45), 3)
```

`ics.get_script_status()`

Accepts a *ics.NeoDevice*, exception on error. Returns a list of values of what ulParameters would hold

`ics.get_serial_number(device)`

Gets the serial number out of the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: Int: Serial Number Version

```
>>> ics.get_serial_number(device)
53123
```

`ics.get_timestamp_for_msg(device, msg)`

Calculates the timestamp for a message.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

msg (*ics.SpyMessage*): *ics.SpyMessage*

Raises: *ics.RuntimeError*

Returns: Float: Timestamp for the message.

```
>>> import ics
>>> d = ics.open_device()
>>> msgs, error_count = ics.get_messages(d)
>>> ics.get_timestamp_for_msg(d, msgs[0])
354577568.9145524
```

`ics.icsneoClosePort()`

Note: Compatibility Function Identical to PEP8 compliant *ics.close_device()* method.

`ics.icsneoEnableBusVoltageMonitor()`

Note: Compatibility Function Identical to PEP8 compliant `ics.enable_bus_voltage_monitor()` method.

`ics.icsneoEnableNetworkCom()`

Note: Compatibility Function Identical to PEP8 compliant `ics.enable_network_com()` method.

`ics.icsneoFindNeoDevices()`

Note: Compatibility Function Identical to PEP8 compliant `ics.find_devices()` method.

`ics.icsneoFirmwareUpdateRequired()`

Note: Compatibility Function Identical to PEP8 compliant `ics.firmware_update_required()` method.

`ics.icsneoForceFirmwareUpdate()`

Note: Compatibility Function Identical to PEP8 compliant `ics.force_firmware_update()` method.

`ics.icsneoGetActiveVNETChannel()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_active_vnet_channel()` method.

`ics.icsneoGetBackupPowerEnabled()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_backup_power_enabled()` method.

`ics.icsneoGetBackupPowerReady()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_backup_power_ready()` method.

`ics.icsneoGetBusVoltage()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_bus_voltage()* method.

`ics.icsneoGetDLLFirmwareInfo()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_dll_firmware_info()* method.

`ics.icsneoGetDLLVersion()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_dll_version()* method.

`ics.icsneoGetDeviceStatus()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_device_status()* method.

`ics.icsneoGetErrorMessages()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_error_messages()* method.

`ics.icsneoGetFireSettings()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_device_settings()* method.

`ics.icsneoGetHWFirmwareInfo()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_hw_firmware_info()* method.

`ics.icsneoGetLastAPIError()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_last_api_error()* method.

`ics.icsneoGetMessages()`

Note: Compatibility Function Identical to PEP8 compliant *ics.get_messages()* method.

`ics.icsneoGetPerformanceParameters()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_performance_parameters()` method.

`ics.icsneoGetRTC()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_rtc()` method.

`ics.icsneoGetSerialNumber()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_serial_number()` method.

`ics.icsneoGetTimeStampForMsg()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_timestamp_for_msg()` method.

`ics.icsneoGetVCAN3Settings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_device_settings()` method.

`ics.icsneoISO15765_DisableNetworks()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_disable_networks()` method.

`ics.icsneoISO15765_EnableNetworks()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_enable_networks()` method.

`ics.icsneoISO15765_ReceiveMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_receive_message()` method.

`ics.icsneoISO15765_TransmitMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.iso15765_transmit_message()` method.

`ics.icsneoLoadDefaultSettings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.load_default_settings()` method.

`ics.icsneoOpenNeoDevice()`

Note: Compatibility Function Identical to PEP8 compliant `ics.open_device()` method.

`ics.icsneoReadSDCard()`

Note: Compatibility Function Identical to PEP8 compliant `ics.read_sdcard()` method.

`ics.icsneoRequestEnterSleepMode()`

Note: Compatibility Function Identical to PEP8 compliant `ics.request_enter_sleep_mode()` method.

`ics.icsneoScriptClear()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_clear()` method.

`ics.icsneoScriptGetFBlockStatus()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_get_fblock_status()` method.

`ics.icsneoScriptGetScriptStatus()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_get_status()` method.

`ics.icsneoScriptGetScriptStatusEx()`

Note: Compatibility Function Identical to PEP8 compliant `ics.get_script_status()` method.

`ics.icsneoScriptLoad()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_load()` method.

`ics.icsneoScriptReadAppSignal()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_read_app_signal()` method.

`ics.icsneoScriptReadRxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_read_rx_message()` method.

`ics.icsneoScriptReadTxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_read_tx_message()` method.

`ics.icsneoScriptStart()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_start()` method.

`ics.icsneoScriptStartFBlock()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_start_fblock()` method.

`ics.icsneoScriptStop()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_stop()` method.

`ics.icsneoScriptStopFBlock()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_stop_fblock()` method.

`ics.icsneoScriptWriteAppSignal()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_write_app_signal()` method.

`ics.icsneoScriptWriteRxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_write_rx_message()` method.

`ics.icsneoScriptWriteTxMessage()`

Note: Compatibility Function Identical to PEP8 compliant `ics.coremini_write_tx_message()` method.

`ics.icsneoSetActiveVNETChannel()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_active_vnet_channel()` method.

`ics.icsneoSetBackupPowerEnabled()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_backup_power_enabled()` method.

`ics.icsneoSetBitRate()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_bit_rate()` method.

`ics.icsneoSetBitRateEx()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_bit_rate_ex()` method.

`ics.icsneoSetContext()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_context()` method.

`ics.icsneoSetFDBitRate()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_fd_bit_rate()` method.

`ics.icsneoSetFireSettings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_device_settings()` method.

`ics.icsneoSetRTC()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_rtc()` method.

`ics.icsneoSetReflashDisplayCallbacks()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_reflash_callback()` method.

`ics.icsneoSetVCAN3Settings()`

Note: Compatibility Function Identical to PEP8 compliant `ics.set_device_settings()` method.

`ics.icsneoTxMessages()`

Note: Compatibility Function Identical to PEP8 compliant `ics.transmit_messages()` method.

`ics.icsneoValidateHObject()`

Note: Compatibility Function Identical to PEP8 compliant `ics.validate_hobject()` method.

`ics.icsneoWriteSDCard()`

Note: Compatibility Function Identical to PEP8 compliant `ics.write_sdcard()` method.

`ics.iso15765_disable_networks(device)`

Disables ISO15765 networks.

Args: `device` (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: None

ics.iso15765_enable_networks (*device, networks*)
Enables ISO15765 networks.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: None

ics.iso15765_receive_message (*device, netid, rx_msg*)
Setup rx ISO15765 Message.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

prx_msg (*ics.CmISO157652RxMessage*): *ics.CmISO157652RxMessage*

Raises: *ics.RuntimeError*

Returns: Boolean: True on success, False on failure.

ics.iso15765_transmit_message (*device, ulNetworkID, pMsg, ulBlockingTimeout*)
Transmits an ISO15765 Message.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

pMsg (*ics.CmISO157652TxMessage*): *ics.CmISO157652TxMessage*

Raises: *ics.RuntimeError*

Returns: Boolean: True on success, False on failure.

ics.load_default_settings (*device*)
Load the default settings in the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: None.

```
>>> device = ics.open_device()
>>> settings = ics.load_default_settings(device)
>>>
```

ics.open_device (*device*)
Opens the device. *device* can be omitted to return a *ics.NeoDevice* of the first free available device, a *ics.NeoDevice*, or a serial number of the device.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

device (int): Serial Number of the device

bNetworkIDs (int): Network Enables

bConfigRead (int): Config Read

iOptions (int): DEVICE_OPTION_* defines

stOptionsOpenNeoEx (int): Usually ics.NETID_CAN, if needed

Raises: *ics.RuntimeError*

Returns: If *ics.NeoDevice* is passed as a parameter, None. If serial number is passed as a parameter, a *ics.NeoDevice* will be returned. If *device* parameter is omitted, a *ics.NeoDevice* will be returned with the first available free device.

```
>>> for device in ics.find_devices():
...     ics.open_device(device)
... 
```

Note: `ics.NeoDevice` will automatically close the device when it goes out of scope.

`ics.override_library_name` (*new_name*)

Overrides the default search for loading the icsneo40 library

Args: name: Absolute path or relative path including filename.

Raises: `ics.RuntimeError`

Returns: None

```
>>> import ics
>>> ics.find_devices()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ics.RuntimeError: Error: find_devices(): Failed to open library: 'icsneo40.dll
↳' with error code: #126
>>> ics.override_library_name(r"C:\Windows\SysWOW64\icsneo40-different.dll")
>>> ics.find_devices()
(<ics.NeoDevice object at 0x00284C50>, <ics.NeoDevice object at 0x007C9A10>)
```

`ics.read_sdcard` ()

icsneoReadSDCard(), Accepts a `ics.NeoDevice` and sector index. Returns a bytearray of 512 bytes max. Exception on error.

`ics.request_enter_sleep_mode` (*device, timeout_ms, mode, reserved_zero*)

Signal neoVI to immediate go to sleep. Currently only supported by FIREVNET/PLASMA. If using over USB this will likely return true but never cause PLASMA to sleep since USB insertion keeps it alive. This API allows Android/Linux applications to invoke power management.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

timeout_ms (int): 16bit word for how long to wait on idle bus before going to sleep. If caller does not want to change it pass in 65535 (0xFFFF) and it will stay whatever it was set to in explorer/coremini.

mode (int): 16bit word for power mode to enter. If caller does not want to change it pass in 65535 (0xFFFF) and it will stay whatever it was set to in explorer/coremini. If it is zero then neoVI will do 'normal sleep'. 0 - power mode off but calling this function will do 'normal sleep'. 1 - normal sleep. 2 - instant sleep. 3 - comatose sleep.

reserved_zero (int): Reserved, Keep as zero.

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

```
>>> ics.request_enter_sleep_mode(device, 1, 0)
True
```

`ics.set_active_vnet_channel` (*device, channel*)

Sets active vnet channel for the device.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

`ics.set_backup_power_enabled(device, enable)`
Sets the device backup power enabled for the device.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

`ics.set_bit_rate(device, BitRate, NetworkID)`
Specifies bit rate setting. Valid values depend on the network specified.

For the networks NETID_HSCAN, NETID_MSCAN, NETID_SWCAN, NETID_FIRE_HSCAN2, NETID_HSCAN3, NETID_LSFTCAN, valid bit rates are 2000, 33333, 50000, 62500, 83333, 100000, 125000, 250000, 500000, 800000, 1000000

For the networks NETID_LIN, NETID_ISO2, NETID_FIRE_LIN2, NETID_FIRE_LIN3, NETID_FIRE_LIN4, valid bit rates are

For the network NETID_FIRE_CGI valid bit rates are 625000 and 115200

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Int: None.

`ics.set_bit_rate_ex(device, BitRate, NetworkID, iOptions)`
Sets the bitrate for a given Network ID on the device with extended options.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Int: None.

`ics.set_context(device)`
Sets the “context” of how `icsneoFindNeoDevices(Ex)` and `icsneoOpenNeoDevice(Ex)` function. If the context is 0 (null) than `icsneoFindNeoDevices(Ex)` will be system wide, searching USB and other supported computer interfaces. `icsneoFindNeoDevices` can then be used to connect to devices found in this manner. If the context is a handle to connected CAN tool than `icsneoFindNeoDevices(Ex)` will search a specific CAN bus for supported IntrepidCS CAN Nodes. Again `icsneoOpenNeoDevice(Ex)` would be used create logical connections to found CAN Nodes.

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

Raises: `ics.RuntimeError`

Returns: Boolean: True on success, False on failure.

```
>>> ics.set_context(device)
True
```

`ics.set_device_settings(device, settings, save_to_eeprom, vnet_slot)`
Sets the settings in the device. `vnet_slot` defaults to `ics.PlasmaIonVnetChannelMain`

Args: device (`ics.NeoDevice`): `ics.NeoDevice`

settings (`ics.DeviceSettings`): `ics.DeviceSettings`

Raises: `ics.RuntimeError`

Returns: None.

```
>>> d = ics.open_device()
>>> d.Name
'neoVI ION'
>>> d.SerialNumber
404444
>>> s = ics.get_device_settings(d, ics.PlasmaIonVnetChannelA) # Get Slave_
↪ settings, channel selection not needed if not a Plasma/Ion
>>> s.DeviceSettingType
2
>>> s.cyan.can_switch_mode
1
>>> s.cyan.can_switch_mode = 2
>>> ics.set_device_settings(d, s, True, ics.PlasmaIonVnetChannelA)
>>>
```

ics.set_fd_bit_rate (*device*, *BitRate*, *NetworkID*)
Sets the FD bitrate for a given Network ID on the device..

Args: *device* (*ics.NeoDevice*): *ics.NeoDevice*

Raises: *ics.RuntimeError*

Returns: Int: None.

ics.set_reflash_callback (*callback*)
Sets the reflash display callback.

Args: *callback* (function): Must be a callable Python function (*def callback(msg, progress)*)

Raises: *ics.RuntimeError*

Returns: None.

```
>>> def callback(msg, progress):
...     print(msg, progress)
...
>>> ics.set_reflash_callback(callback)
>>>
```

ics.set_rtc (*device*[, *time*])
Sets the Real-Time Clock of the device.

Args: *device* (*ics.NeoDevice*): *ics.NeoDevice*

ime (datetime.datetime): Optional. Sets to current time, if omitted.

Raises: *ics.RuntimeError*

Returns: None.

```
>>> device = ics.open_device()
>>> ics.set_rtc(device)
```

ics.transmit_messages (*device*, *messages*)
Transmits message(s) on the device. *messages* can be a tuple of *ics.SpyMessage*

Args: *device* (*ics.NeoDevice*): *ics.NeoDevice*

messages (*ics.SpyMessage*): *ics.SpyMessage*

Raises: *ics.RuntimeError*

Returns: None.


```
>>> device = ics.open_device()
>>> msg = ics.SpyMessage()
>>> msg.ArbIDOrHeader = 0xFF
>>> msg.NetworkID = ics.NETID_HSCAN
>>> msg.Data = (0,1,2,3,4,5,6,7)
>>> ics.transmit_messages(device, msg)
>>>
```

ics.validate_hobject (*device*)

Validates the handle is valid for a *device*. Handles are only valid when the device is open.

Args: device (*ics.NeoDevice*): *ics.NeoDevice*

or:

device (int): c style integer handle to the device.

Raises: *ics.RuntimeError*

Returns: Boolean: True if valid, false otherwise.

```
>>> device = ics.open_device()
>>> ics.validate_hobject(device)
1
>>> ics.validate_hobject(device._Handle)
1
```

ics.write_sdcard ()

icsneoReadSDCard(), Accepts a ics.NeoDevice, sector index, and a bytearray of 512 bytes. Exception on error.

CHAPTER 7

Module Variables

```
ics.AUTO = 0
ics.BPS100 = 5
ics.BPS1000 = 10
ics.BPS100000 = 7
ics.BPS10400 = 1
ics.BPS117647 = 8
ics.BPS125 = 6
ics.BPS20 = 0
ics.BPS2000 = 12
ics.BPS250 = 7
ics.BPS33 = 1
ics.BPS33333 = 2
ics.BPS4000 = 13
ics.BPS50 = 2
ics.BPS500 = 8
ics.BPS5000 = 0
ics.BPS50000 = 3
ics.BPS62 = 3
ics.BPS62500 = 4
ics.BPS666 = 11
ics.BPS71429 = 5
ics.BPS800 = 9
```

```
ics.BPS83 = 4
ics.BPS83333 = 6
ics.BUILD_DATETIME = May 29 2018 16:04:26
ics.CANFD_BRS_ENABLED = 2
ics.CANFD_BRS_ENABLED_ISO = 4
ics.CANFD_ENABLED = 1
ics.CANFD_ENABLED_ISO = 3
ics.CANFD_SETTINGS_SIZE = 10
ics.CANTERM_SETTINGS_SIZE = 6
ics.CAN_BPS10000 = 17
ics.CAN_BPS5000 = 14
ics.CAN_BPS6667 = 15
ics.CAN_BPS8000 = 16
ics.CAN_SETTINGS_SIZE = 12
ics.DISABLE = 1
ics.ETHERNET_SETTINGS_SIZE = 8
ics.FAST_MODE = 3
ics.GLOBAL_SETTINGS_SIZE = 908
ics.GS_VERSION = 5
ics.ISO15765_2_NETWORK_HSCAN = 1
ics.ISO15765_2_NETWORK_HSCAN2 = 4
ics.ISO15765_2_NETWORK_HSCAN3 = 8
ics.ISO15765_2_NETWORK_HSCAN4 = 20
ics.ISO15765_2_NETWORK_HSCAN5 = 24
ics.ISO15765_2_NETWORK_HSCAN6 = 28
ics.ISO15765_2_NETWORK_HSCAN7 = 32
ics.ISO15765_2_NETWORK_MSCAN = 2
ics.ISO15765_2_NETWORK_SWCAN = 16
ics.ISO15765_2_NETWORK_SWCAN2 = 36
ics.ISO9141_KEYWORD2000_SETTINGS_SIZE = 114
ics.ISO9141_KEYWORD2000__INIT_STEP_SIZE = 6
ics.J1708_SETTINGS_SIZE = 2
ics.LIN_SETTINGS_SIZE = 10
ics.LISTEN_ALL = 7
ics.LISTEN_ONLY = 3
ics.LOOPBACK = 2
```

```
ics.NEODEVICE_ALL = -16385
ics.NEODEVICE_ANY_ION = 1310720
ics.NEODEVICE_ANY_PLASMA = 208896
ics.NEODEVICE_BLUE = 1
ics.NEODEVICE_CMPROBE = 8388608
ics.NEODEVICE_CT_OBD = 32768
ics.NEODEVICE_DW_VCAN = 4
ics.NEODEVICE_ECU = 128
ics.NEODEVICE_ECUCHIP_UART = 2048
ics.NEODEVICE_EEVB = 16777216
ics.NEODEVICE_FIRE = 8
ics.NEODEVICE_FIRE2 = 67108864
ics.NEODEVICE_FIRE_VNET = 8192
ics.NEODEVICE_FLEX = 134217728
ics.NEODEVICE_IEVB = 256
ics.NEODEVICE_ION_2 = 262144
ics.NEODEVICE_ION_3 = 1048576
ics.NEODEVICE_NEOANALOG = 16384
ics.NEODEVICE_NEOECUCHIP = 256
ics.NEODEVICE_OBD2_PRO = 1024
ics.NEODEVICE_OBD2_SIM = -2147483648
ics.NEODEVICE_PENDANT = 512
ics.NEODEVICE_PLASMA_1_11 = 4096
ics.NEODEVICE_PLASMA_1_12 = 65536
ics.NEODEVICE_PLASMA_1_13 = 131072
ics.NEODEVICE_RADGALAXY = 268435456
ics.NEODEVICE_RADSTAR = 524288
ics.NEODEVICE_RADSTAR2 = 536870912
ics.NEODEVICE_RED = 64
ics.NEODEVICE_SW_VCAN = 2
ics.NEODEVICE_UNKNOWN = 0
ics.NEODEVICE_VCAN3 = 16
ics.NEODEVICE_VCAN4 = 2097152
ics.NEODEVICE_VCAN4_12 = 4194304
ics.NEODEVICE_VCANRF = 33554432
ics.NEODEVICE_VIVIDCAN = 1073741824
```

```
ics.NEOVI6_VCAN_TIMESTAMP_1 = 1e-06
ics.NEOVI6_VCAN_TIMESTAMP_2 = 0.065536
ics.NEOVIPRO_VCAN_TIMESTAMP_1 = 1e-06
ics.NEOVIPRO_VCAN_TIMESTAMP_2 = 0.065536
ics.NEOVI_3G_MAX_SETTINGS_SIZE = 908
ics.NEOVI_COMMTYPE_FIRE_USB = 5
ics.NEOVI_COMMTYPE_RS232 = 0
ics.NEOVI_COMMTYPE_TCPIP = 3
ics.NEOVI_COMMTYPE_USB_BULK = 1
ics.NEOVI_RED_TIMESTAMP_1_10NS = 1e-08
ics.NEOVI_RED_TIMESTAMP_1_25NS = 2.5e-08
ics.NEOVI_RED_TIMESTAMP_2_10NS = 429.4967296
ics.NEOVI_RED_TIMESTAMP_2_25NS = 107.3741824
ics.NEOVI_TIMESTAMP_1 = 1.6e-06
ics.NEOVI_TIMESTAMP_2 = 0.1048576
ics.NEO_CFG_MPIC_HS_CAN_CNF1 = 522
ics.NEO_CFG_MPIC_HS_CAN_CNF2 = 521
ics.NEO_CFG_MPIC_HS_CAN_CNF3 = 520
ics.NEO_CFG_MPIC_HS_CAN_MODE = 566
ics.NEO_CFG_MPIC_LSFT_CAN_CNF1 = 558
ics.NEO_CFG_MPIC_LSFT_CAN_CNF2 = 557
ics.NEO_CFG_MPIC_LSFT_CAN_CNF3 = 556
ics.NEO_CFG_MPIC_MS_CAN_CNF1 = 534
ics.NEO_CFG_MPIC_MS_CAN_CNF2 = 533
ics.NEO_CFG_MPIC_MS_CAN_CNF3 = 532
ics.NEO_CFG_MPIC_SW_CAN_CNF1 = 546
ics.NEO_CFG_MPIC_SW_CAN_CNF2 = 545
ics.NEO_CFG_MPIC_SW_CAN_CNF3 = 544
ics.NETID_3G_APP_SIGNAL_STATUS = 56
ics.NETID_3G_FB_STATUS = 55
ics.NETID_3G_LOGGING_OVERFLOW = 59
ics.NETID_3G_READ_DATA LINK_CM_RX_MSG = 58
ics.NETID_3G_READ_DATA LINK_CM_TX_MSG = 57
ics.NETID_3G_READ_SETTINGS_EX = 60
ics.NETID_3G_RESET_STATUS = 54
ics.NETID_AUX = 7
```

```
ics.NETID_CGI = 53
ics.NETID_DATA_TO_HOST = 70
ics.NETID_DEVICE = 0
ics.NETID_DEVICE_STATUS = 513
ics.NETID_ETHERNET = 93
ics.NETID_ETHERNET_DAQ = 69
ics.NETID_FLEXRAY = 85
ics.NETID_FLEXRAY1A = 80
ics.NETID_FLEXRAY1B = 81
ics.NETID_FLEXRAY2 = 86
ics.NETID_FLEXRAY2A = 82
ics.NETID_FLEXRAY2B = 83
ics.NETID_FORDSCP = 5
ics.NETID_GMFSA = 94
ics.NETID_HSCAN = 1
ics.NETID_HSCAN2 = 42
ics.NETID_HSCAN3 = 44
ics.NETID_HSCAN4 = 61
ics.NETID_HSCAN5 = 62
ics.NETID_HSCAN6 = 96
ics.NETID_HSCAN7 = 97
ics.NETID_HW_COM_LATENCY_TEST = 512
ics.NETID_I2C1 = 71
ics.NETID_INVALID = 65535
ics.NETID_ISO = 9
ics.NETID_ISO14230 = 15
ics.NETID_ISO2 = 14
ics.NETID_ISO3 = 41
ics.NETID_ISO4 = 47
ics.NETID_ISOPIC = 10
ics.NETID_J1708 = 6
ics.NETID_JVPW = 8
ics.NETID_LIN = 16
ics.NETID_LIN2 = 48
ics.NETID_LIN3 = 49
ics.NETID_LIN4 = 50
```

```
ics.NETID_LIN5 = 84
ics.NETID_LIN6 = 98
ics.NETID_LSFTCAN = 4
ics.NETID_LSFTCAN2 = 99
ics.NETID_MAIN51 = 11
ics.NETID_MAX = 100
ics.NETID_MOST = 51
ics.NETID_MOST150 = 92
ics.NETID_MOST25 = 90
ics.NETID_MOST50 = 91
ics.NETID_MSCAN = 2
ics.NETID_OP_ETHERNET1 = 17
ics.NETID_OP_ETHERNET10 = 78
ics.NETID_OP_ETHERNET11 = 79
ics.NETID_OP_ETHERNET12 = 87
ics.NETID_OP_ETHERNET2 = 18
ics.NETID_OP_ETHERNET3 = 19
ics.NETID_OP_ETHERNET4 = 45
ics.NETID_OP_ETHERNET5 = 46
ics.NETID_OP_ETHERNET6 = 73
ics.NETID_OP_ETHERNET7 = 75
ics.NETID_OP_ETHERNET8 = 76
ics.NETID_OP_ETHERNET9 = 77
ics.NETID_RED = 12
ics.NETID_RED_APP_ERROR = 52
ics.NETID_RED_VBAT = 74
ics.NETID_RS232 = 63
ics.NETID_SCI = 13
ics.NETID_SPI1 = 72
ics.NETID_SWCAN = 3
ics.NETID_SWCAN2 = 68
ics.NETID_TCP = 95
ics.NETID_TEXTAPI_TO_HOST = 71
ics.NETID_UART = 64
ics.NETID_UART2 = 65
ics.NETID_UART3 = 66
```



```
ics.NETID_UART4 = 67
ics.NORMAL = 0
ics.NORMAL_MODE = 2
ics.NO_CANFD = 0
ics.OPETH_FUNC_MEDIACONVERTER = 1
ics.OPETH_FUNC_TAP = 0
ics.OPETH_FUNC_TAP_LOW_LATENCY = 2
ics.OPETH_LINK_AUTO = 0
ics.OPETH_LINK_MASTER = 1
ics.OPETH_LINK_SLAVE = 2
ics.OPETH_MAC_SPOOF_DST_ADDR = 0
ics.OPETH_MAC_SPOOF_SRC_ADDR = 1
ics.OP_ETH_GENERAL_SETTINGS_SIZE = 20
ics.OP_ETH_SETTINGS_SIZE = 16
ics.PLASMA_SLAVE1_OFFSET = 100
ics.PLASMA_SLAVE1_OFFSET_RANGE2 = 4608
ics.PLASMA_SLAVE2_OFFSET = 200
ics.PLASMA_SLAVE2_OFFSET_RANGE2 = 8704
ics.PLASMA_SLAVE3_OFFSET_RANGE2 = 12800
ics.PLASMA_SLAVE_NUM = 51
ics.REPORT_ON_GPS = 15
ics.REPORT_ON_KLINE = 9
ics.REPORT_ON_LED1 = 7
ics.REPORT_ON_LED2 = 8
ics.REPORT_ON_MISC1 = 1
ics.REPORT_ON_MISC2 = 2
ics.REPORT_ON_MISC3 = 3
ics.REPORT_ON_MISC3_AIN = 10
ics.REPORT_ON_MISC4 = 4
ics.REPORT_ON_MISC4_AIN = 11
ics.REPORT_ON_MISC5 = 5
ics.REPORT_ON_MISC5_AIN = 12
ics.REPORT_ON_MISC6 = 6
ics.REPORT_ON_MISC6_AIN = 13
ics.REPORT_ON_PERIODIC = 0
ics.REPORT_ON_PWM_IN1 = 14
```

```
ics.RESISTOR_OFF = 1
ics.RESISTOR_ON = 0
ics.SCRIPT_LOCATION_FLASH_MEM = 0
ics.SCRIPT_LOCATION_INTERNAL_FLASH = 2
ics.SCRIPT_LOCATION_SDCARD = 1
ics.SCRIPT_LOCATION_VCAN3_MEM = 4
ics.SCRIPT_STATUS_RUNNING = 1
ics.SCRIPT_STATUS_STOPPED = 0
ics.SLEEP_MODE = 0
ics.SLOW_MODE = 1
ics.SPY_PROTOCOL_BEAN = 11
ics.SPY_PROTOCOL_CAN = 1
ics.SPY_PROTOCOL_CANFD = 30
ics.SPY_PROTOCOL_CGI = 18
ics.SPY_PROTOCOL_CHRYSLER_CCD = 8
ics.SPY_PROTOCOL_CHRYSLER_JVPW = 14
ics.SPY_PROTOCOL_CHRYSLER_SCI = 9
ics.SPY_PROTOCOL_CUSTOM = 0
ics.SPY_PROTOCOL_DALLAS_1WIRE = 25
ics.SPY_PROTOCOL_ETHERNET = 29
ics.SPY_PROTOCOL_FLEXRAY = 16
ics.SPY_PROTOCOL_FORD_UBP = 10
ics.SPY_PROTOCOL_GENERIC_MANCHSESTER = 26
ics.SPY_PROTOCOL_GENERIC_UART = 22
ics.SPY_PROTOCOL_GME_CIM_SCL_KLINE = 19
ics.SPY_PROTOCOL_GMFSA = 31
ics.SPY_PROTOCOL_GMLAN = 2
ics.SPY_PROTOCOL_GM_ALDL_UART = 7
ics.SPY_PROTOCOL_I2C = 21
ics.SPY_PROTOCOL_ISO9141 = 5
ics.SPY_PROTOCOL_J1708 = 13
ics.SPY_PROTOCOL_J1850PWM = 4
ics.SPY_PROTOCOL_J1850VPW = 3
ics.SPY_PROTOCOL_J1939 = 15
ics.SPY_PROTOCOL_JTAG = 23
ics.SPY_PROTOCOL_LIN = 12
```

```
ics.SPY_PROTOCOL_MOST = 17
ics.SPY_PROTOCOL_SENT_PROTOCOL = 27
ics.SPY_PROTOCOL_SPI = 20
ics.SPY_PROTOCOL_TCP = 32
ics.SPY_PROTOCOL_UART = 28
ics.SPY_PROTOCOL_UNIO = 24
ics.SPY_STATUS2_CAN_HAVE_LINK_DATA = 4194304
ics.SPY_STATUS2_CAN_ISO15765_LOGICAL_FRAME = 2097152
ics.SPY_STATUS2_END_OF_LONG_MESSAGE = 1048576
ics.SPY_STATUS2_ERROR_FRAME = 131072
ics.SPY_STATUS2_ETHERNET_CRC_ERROR = 2097152
ics.SPY_STATUS2_ETHERNET_FCS_AVAILABLE = 8388608
ics.SPY_STATUS2_ETHERNET_FRAME_TOO_SHORT = 4194304
ics.SPY_STATUS2_ETHERNET_NO_PADDING = 16777216
ics.SPY_STATUS2_ETHERNET_PREEMPTION_ENABLED = 33554432
ics.SPY_STATUS2_FLEXRAY_NO_CRC = 33554432
ics.SPY_STATUS2_FLEXRAY_NO_HEADERCRC = 67108864
ics.SPY_STATUS2_FLEXRAY_TX_AB = 2097152
ics.SPY_STATUS2_FLEXRAY_TX_AB_NO_A = 4194304
ics.SPY_STATUS2_FLEXRAY_TX_AB_NO_B = 8388608
ics.SPY_STATUS2_FLEXRAY_TX_AB_NO_MATCH = 16777216
ics.SPY_STATUS2_GLOBAL_CHANGE = 65536
ics.SPY_STATUS2_HAS_VALUE = 1
ics.SPY_STATUS2_HIGH_VOLTAGE = 4
ics.SPY_STATUS2_ISO_FRAME_ERROR = 134217728
ics.SPY_STATUS2_ISO_OVERFLOW_ERROR = 268435456
ics.SPY_STATUS2_ISO_PARITY_ERROR = 536870912
ics.SPY_STATUS2_LIN_ERR_MSG_ID_PARITY = 67108864
ics.SPY_STATUS2_LIN_ERR_RX_BREAK_NOT_0 = 2097152
ics.SPY_STATUS2_LIN_ERR_RX_BREAK_TOO_SHORT = 4194304
ics.SPY_STATUS2_LIN_ERR_RX_DATA_GREATER_8 = 16777216
ics.SPY_STATUS2_LIN_ERR_RX_SYNC_NOT_55 = 8388608
ics.SPY_STATUS2_LIN_ERR_TX_RX_MISMATCH = 33554432
ics.SPY_STATUS2_LIN_ID_FRAME_ERROR = 268435456
ics.SPY_STATUS2_LIN_NO_SLAVE_DATA = -2147483648
ics.SPY_STATUS2_LIN_SLAVE_BYTE_ERROR = 536870912
```

```
ics.SPY_STATUS2_LIN_SYNC_FRAME_ERROR = 134217728
ics.SPY_STATUS2_LONG_MESSAGE = 8
ics.SPY_STATUS2_MOST_CHANGED_PAR = -2147483648
ics.SPY_STATUS2_MOST_CONTROL_DATA = 16777216
ics.SPY_STATUS2_MOST_I2S_DUMP = 134217728
ics.SPY_STATUS2_MOST_LOW_LEVEL = 8388608
ics.SPY_STATUS2_MOST_MHP_CONTROL_DATA = 67108864
ics.SPY_STATUS2_MOST_MHP_USER_DATA = 33554432
ics.SPY_STATUS2_MOST_MOST150 = 1073741824
ics.SPY_STATUS2_MOST_MOST50 = 536870912
ics.SPY_STATUS2_MOST_PACKET_DATA = 2097152
ics.SPY_STATUS2_MOST_TOO_SHORT = 268435456
ics.SPY_STATUS2_RX_TIMEOUT_ERROR = 1073741824
ics.SPY_STATUS2_VALUE_IS_BOOLEAN = 2
ics.SPY_STATUS3_CANFD_BRS = 16
ics.SPY_STATUS3_CANFD_ESI = 1
ics.SPY_STATUS3_CANFD_FDF = 8
ics.SPY_STATUS3_CANFD_IDE = 2
ics.SPY_STATUS3_CANFD_RTR = 4
ics.SPY_STATUS3_LIN_JUST_BREAK_SYNC = 1
ics.SPY_STATUS3_LIN_ONLY_UPDATE_SLAVE_TABLE_ONCE = 4
ics.SPY_STATUS3_LIN_SLAVE_DATA_TOO_SHORT = 2
ics.SPY_STATUS_ANALOG_DIGITAL_INPUT = 16777216
ics.SPY_STATUS_AUDIO_COMMENT = 4194304
ics.SPY_STATUS_AVSI_REC_OVERFLOW = 1048576
ics.SPY_STATUS_BAD_MESSAGE_BIT_TIME_ERROR = 16384
ics.SPY_STATUS_BREAK = 524288
ics.SPY_STATUS_BUS_RECOVERED = 1024
ics.SPY_STATUS_BUS_SHORTED_GND = 4096
ics.SPY_STATUS_BUS_SHORTED_PLUS = 2048
ics.SPY_STATUS_CANFD = 536870912
ics.SPY_STATUS_CAN_BUS_OFF = 512
ics.SPY_STATUS_CAN_ERROR_PASSIVE = 32
ics.SPY_STATUS_CHECKSUM_ERROR = 8192
ics.SPY_STATUS_COMM_IN_OVERFLOW = 65536
ics.SPY_STATUS_CRC_ERROR = 16
```

```
ics.SPY_STATUS_EXPECTED_LEN_MISMATCH = 131072
ics.SPY_STATUS_EXTENDED = -2147483648
ics.SPY_STATUS_FLEXRAY_PDU = 536870912
ics.SPY_STATUS_FLEXRAY_PDU_NO_UPDATE_BIT = 8
ics.SPY_STATUS_FLEXRAY_PDU_UPDATE_BIT_SET = 1073741824
ics.SPY_STATUS_GLOBAL_ERR = 1
ics.SPY_STATUS_GPS_DATA = 8388608
ics.SPY_STATUS_HEADERCRC_ERROR = 32
ics.SPY_STATUS_HIGH_SPEED = 1073741824
ics.SPY_STATUS_INCOMPLETE_FRAME = 64
ics.SPY_STATUS_INIT_MESSAGE = 536870912
ics.SPY_STATUS_LIN_MASTER = 536870912
ics.SPY_STATUS_LOST_ARBITRATION = 128
ics.SPY_STATUS_MSG_NO_MATCH = 262144
ics.SPY_STATUS_NETWORK_MESSAGE_TYPE = 67108864
ics.SPY_STATUS_PDU = 536870912
ics.SPY_STATUS_REMOTE_FRAME = 8
ics.SPY_STATUS_TEST_TRIGGER = 2097152
ics.SPY_STATUS_TEXT_COMMENT = 33554432
ics.SPY_STATUS_TX_MSG = 2
ics.SPY_STATUS_TX_NOMATCH = 32768
ics.SPY_STATUS_UNDEFINED_ERROR = 256
ics.SPY_STATUS_VSI_IFR_CRC_BIT = 268435456
ics.SPY_STATUS_VSI_TX_UNDERRUN = 134217728
ics.SPY_STATUS_XTD_FRAME = 4
ics.SWCAN_AUTOSWITCH_DISABLED = 0
ics.SWCAN_AUTOSWITCH_DISABLED_RESISTOR_ENABLED = 3
ics.SWCAN_AUTOSWITCH_NO_RESISTOR = 1
ics.SWCAN_AUTOSWITCH_WITH_RESISTOR = 2
ics.SWCAN_SETTINGS_SIZE = 14
ics.UART_SETTINGS_SIZE = 16
ics.USE_TQ = 1
ics.VNETBITS_FEATURE_ANDROID_MSGS = 1
ics.VNETBITS_FEATURE_DISABLE_USB_CHECK = 2
```


i

ics, [19](#)

A

AckBytes (*ics.SpyMessage* attribute), 34
 AckBytes (*ics.SpyMessageJ1850* attribute), 35
 ain_sample_period (*ics.CyanSettings* attribute), 23
 ain_sample_period (*ics.FireSettings* attribute), 26
 ain_sample_period (*ics.RadGalaxySettings* attribute), 30
 ain_sample_period (*ics.RadStar2Settings* attribute), 32
 ain_threshold (*ics.CyanSettings* attribute), 23
 ain_threshold (*ics.FireSettings* attribute), 26
 ain_threshold (*ics.RadGalaxySettings* attribute), 30
 ain_threshold (*ics.RadStar2Settings* attribute), 32
 ApiFirmwareInfo (class in *ics*), 20
 ArbIDOrHeader (*ics.SpyMessage* attribute), 34
 ArgumentError, 19
 AUTO (in module *ics*), 71
 auto_baud (*ics.CanSettings* attribute), 21
 auto_baud (*ics.SWCanSettings* attribute), 34
 auto_neg (*ics.EthernetSettings* attribute), 26
 AutoHandleClose (*ics.NeoDevice* attribute), 29

B

backupPowerEnabled (*ics.Fire2DeviceStatus* attribute), 26
 backupPowerGood (*ics.Fire2DeviceStatus* attribute), 26
 base36enc () (in module *ics*), 48
 Baudrate (*ics.CanSettings* attribute), 21
 Baudrate (*ics.Iso9141Keyword2000Settings* attribute), 28
 Baudrate (*ics.LinSettings* attribute), 29
 Baudrate (*ics.SWCanSettings* attribute), 33
 Baudrate (*ics.UartSettings* attribute), 37
 bEnReportLinkQuality (*ics.OpEthGeneralSettings* attribute), 29
 blockSize (*ics.CmISO157652RxMessage* attribute), 21

blockSize (*ics.CmISO157652TxMessage* attribute), 22
 bOptions (*ics.UartSettings* attribute), 37
 BPS100 (in module *ics*), 71
 BPS1000 (in module *ics*), 71
 BPS100000 (in module *ics*), 71
 BPS10400 (in module *ics*), 71
 BPS117647 (in module *ics*), 71
 BPS125 (in module *ics*), 71
 BPS20 (in module *ics*), 71
 BPS2000 (in module *ics*), 71
 BPS250 (in module *ics*), 71
 BPS33 (in module *ics*), 71
 BPS33333 (in module *ics*), 71
 BPS4000 (in module *ics*), 71
 BPS50 (in module *ics*), 71
 BPS500 (in module *ics*), 71
 BPS5000 (in module *ics*), 71
 BPS50000 (in module *ics*), 71
 BPS62 (in module *ics*), 71
 BPS62500 (in module *ics*), 71
 BPS666 (in module *ics*), 71
 BPS71429 (in module *ics*), 71
 BPS800 (in module *ics*), 71
 BPS83 (in module *ics*), 71
 BPS83333 (in module *ics*), 72
 brgh (*ics.Iso9141Keyword2000Settings* attribute), 28
 brgh (*ics.LinSettings* attribute), 29
 brgh (*ics.UartSettings* attribute), 37
 BRP (*ics.CanSettings* attribute), 21
 BRP (*ics.SWCanSettings* attribute), 33
 bTapEnPtp (*ics.OpEthGeneralSettings* attribute), 29
 bTapEnSwitch (*ics.OpEthGeneralSettings* attribute), 29
 BUILD_DATETIME (in module *ics*), 72

C

can1 (*ics.CyanSettings* attribute), 23
 can1 (*ics.FireSettings* attribute), 26
 can1 (*ics.RadGalaxySettings* attribute), 30

can1 (*ics.RadStar2Settings* attribute), 32
can1 (*ics.Vcan3Settings* attribute), 37
can1 (*ics.Vcan412Settings* attribute), 38
can1 (*ics.Vcan4Settings* attribute), 38
can1 (*ics.VcanRFSettings* attribute), 39
can1 (*ics.VividCANSettings* attribute), 40
can1_options (*ics.TextApiSettings* attribute), 36
can1_rx_id (*ics.TextApiSettings* attribute), 36
can1_tx_id (*ics.TextApiSettings* attribute), 36
can2 (*ics.CyanSettings* attribute), 23
can2 (*ics.FireSettings* attribute), 26
can2 (*ics.RadGalaxySettings* attribute), 30
can2 (*ics.RadStar2Settings* attribute), 32
can2 (*ics.Vcan3Settings* attribute), 37
can2 (*ics.Vcan412Settings* attribute), 38
can2 (*ics.Vcan4Settings* attribute), 38
can2 (*ics.VcanRFSettings* attribute), 39
can2_options (*ics.TextApiSettings* attribute), 36
can2_rx_id (*ics.TextApiSettings* attribute), 36
can2_tx_id (*ics.TextApiSettings* attribute), 36
can3 (*ics.CyanSettings* attribute), 23
can3 (*ics.FireSettings* attribute), 26
can3 (*ics.RadGalaxySettings* attribute), 30
can3 (*ics.Vcan4Settings* attribute), 38
can3 (*ics.VcanRFSettings* attribute), 39
can3_options (*ics.TextApiSettings* attribute), 36
can3_rx_id (*ics.TextApiSettings* attribute), 36
can3_tx_id (*ics.TextApiSettings* attribute), 36
can4 (*ics.CyanSettings* attribute), 23
can4 (*ics.FireSettings* attribute), 26
can4 (*ics.RadGalaxySettings* attribute), 30
can4 (*ics.Vcan4Settings* attribute), 38
can4 (*ics.VcanRFSettings* attribute), 40
can4_options (*ics.TextApiSettings* attribute), 36
can4_rx_id (*ics.TextApiSettings* attribute), 36
can4_tx_id (*ics.TextApiSettings* attribute), 36
can5 (*ics.CyanSettings* attribute), 23
can5 (*ics.RadGalaxySettings* attribute), 30
can6 (*ics.CyanSettings* attribute), 23
can6 (*ics.RadGalaxySettings* attribute), 30
can7 (*ics.CyanSettings* attribute), 23
can7 (*ics.RadGalaxySettings* attribute), 30
can8 (*ics.CyanSettings* attribute), 23
can8 (*ics.RadGalaxySettings* attribute), 30
CAN_BPS10000 (*in module ics*), 72
CAN_BPS5000 (*in module ics*), 72
CAN_BPS6667 (*in module ics*), 72
CAN_BPS8000 (*in module ics*), 72
CAN_SETTINGS_SIZE (*in module ics*), 72
can_switch_mode (*ics.CyanSettings* attribute), 23
can_switch_mode (*ics.RadGalaxySettings* attribute), 30
can_switch_mode (*ics.RadStar2Settings* attribute), 32
can_switch_mode (*ics.VividCANSettings* attribute), 41
canfd1 (*ics.CyanSettings* attribute), 23
canfd1 (*ics.RadGalaxySettings* attribute), 30
canfd1 (*ics.RadStar2Settings* attribute), 32
canfd1 (*ics.Vcan412Settings* attribute), 38
canfd1 (*ics.Vcan4Settings* attribute), 38
canfd2 (*ics.CyanSettings* attribute), 23
canfd2 (*ics.RadGalaxySettings* attribute), 30
canfd2 (*ics.RadStar2Settings* attribute), 32
canfd2 (*ics.Vcan412Settings* attribute), 38
canfd2 (*ics.Vcan4Settings* attribute), 39
canfd3 (*ics.CyanSettings* attribute), 23
canfd3 (*ics.RadGalaxySettings* attribute), 30
canfd3 (*ics.Vcan4Settings* attribute), 39
canfd4 (*ics.CyanSettings* attribute), 23
canfd4 (*ics.RadGalaxySettings* attribute), 30
canfd4 (*ics.Vcan4Settings* attribute), 39
canfd5 (*ics.CyanSettings* attribute), 23
canfd5 (*ics.RadGalaxySettings* attribute), 30
canfd6 (*ics.CyanSettings* attribute), 23
canfd6 (*ics.RadGalaxySettings* attribute), 30
canfd7 (*ics.CyanSettings* attribute), 23
canfd7 (*ics.RadGalaxySettings* attribute), 30
canfd8 (*ics.CyanSettings* attribute), 23
canfd8 (*ics.RadGalaxySettings* attribute), 30
CANFD_BRS_ENABLED (*in module ics*), 72
CANFD_BRS_ENABLED_ISO (*in module ics*), 72
CANFD_ENABLED (*in module ics*), 72
CANFD_ENABLED_ISO (*in module ics*), 72
CANFD_SETTINGS_SIZE (*in module ics*), 72
CanFdSettings (*class in ics*), 20
CanSettings (*class in ics*), 20
CANTERM_SETTINGS_SIZE (*in module ics*), 72
cf_timeout (*ics.CmISO157652RxMessage* attribute), 21
cgi_baud (*ics.FireSettings* attribute), 26
cgi_chksum_enable (*ics.FireSettings* attribute), 26
cgi_enable_reserved (*ics.FireSettings* attribute), 26
cgi_rx_ifs_bit_times (*ics.FireSettings* attribute), 26
cgi_tx_ifs_bit_times (*ics.FireSettings* attribute), 26
chksum_enabled (*ics.Iso9141Keyword2000Settings* attribute), 28
close_device() (*in module ics*), 48
ClosePort() (*in module ics*), 41
CmISO157652RxMessage (*class in ics*), 21
CmISO157652TxMessage (*class in ics*), 22
coremini_clear() (*in module ics*), 48
coremini_get_fblock_status() (*in module ics*), 49
coremini_get_status() (*in module ics*), 49

coremini_load() (in module ics), 49
 coremini_read_app_signal() (in module ics), 49
 coremini_read_rx_message() (in module ics), 50
 coremini_read_tx_message() (in module ics), 50
 coremini_start() (in module ics), 50
 coremini_start_fblock() (in module ics), 50
 coremini_stop() (in module ics), 51
 coremini_stop_fblock() (in module ics), 51
 coremini_write_app_signal() (in module ics), 51
 coremini_write_rx_message() (in module ics), 51
 coremini_write_tx_message() (in module ics), 51
 create_neovi_radio_message() (in module ics), 51
 cyan (ics.DeviceSettings attribute), 25
 CyanSettings (class in ics), 22

D

data (ics.CmISO157652TxMessage attribute), 22
 Data (ics.SpyMessage attribute), 34
 Data (ics.SpyMessageJ1850 attribute), 35
 DescriptionID (ics.SpyMessage attribute), 34
 DescriptionID (ics.SpyMessageJ1850 attribute), 35
 DeviceSettings (class in ics), 25
 DeviceSettingType (ics.DeviceSettings attribute), 25
 DeviceType (ics.NeoDevice attribute), 29
 digitalIoThresholdEnable (ics.CyanSettings attribute), 23
 digitalIoThresholdTicks (ics.CyanSettings attribute), 23
 DISABLE (in module ics), 72
 disableUsbCheckOnBoot (ics.CyanSettings attribute), 23
 disableUsbCheckOnBoot (ics.Vcan412Settings attribute), 38
 disableUsbCheckOnBoot (ics.VividCANSettings attribute), 41
 duplex (ics.EthernetSettings attribute), 26

E

ecu_id (ics.VividCANSettings attribute), 41
 enable_bus_voltage_monitor() (in module ics), 52
 enable_network_com() (in module ics), 52
 EnableBusVoltageMonitor() (in module ics), 41
 enableLatencyTest (ics.CyanSettings attribute), 23
 enableLatencyTest (ics.Vcan412Settings attribute), 38

enableLatencyTest (ics.Vcan4Settings attribute), 39
 enableLatencyTest (ics.VividCANSettings attribute), 41
 EnableNetworkCom() (in module ics), 41
 enablePcEthernetComm (ics.Vcan4Settings attribute), 39
 ethernet (ics.CyanSettings attribute), 23
 ethernet (ics.Vcan4Settings attribute), 39
 ETHERNET_SETTINGS_SIZE (in module ics), 72
 ethernetActivationLineEnabled (ics.Fire2DeviceStatus attribute), 26
 ethernetActivationLineEnabled (ics.Vcan4DeviceStatus attribute), 38
 EthernetSettings (class in ics), 26
 extendedAddress (ics.CmISO157652RxMessage attribute), 21
 extendedAddress (ics.CmISO157652TxMessage attribute), 22
 ExtraDataPtr (ics.SpyMessage attribute), 34
 ExtraDataPtr (ics.SpyMessageJ1850 attribute), 35
 ExtraDataPtrEnabled (ics.SpyMessage attribute), 34
 ExtraDataPtrEnabled (ics.SpyMessageJ1850 attribute), 35

F

fast_init_network_enables_1 (ics.FireSettings attribute), 26
 fast_init_network_enables_2 (ics.FireSettings attribute), 26
 FAST_MODE (in module ics), 72
 fc_id (ics.CmISO157652RxMessage attribute), 21
 fc_id (ics.CmISO157652TxMessage attribute), 22
 fc_id_mask (ics.CmISO157652TxMessage attribute), 22
 FDBaudrate (ics.CanFdSettings attribute), 20
 FDBRP (ics.CanFdSettings attribute), 20
 FDMODE (ics.CanFdSettings attribute), 20
 FDTqProp (ics.CanFdSettings attribute), 20
 FDTqSeg1 (ics.CanFdSettings attribute), 20
 FDTqSeg2 (ics.CanFdSettings attribute), 20
 FDTqSync (ics.CanFdSettings attribute), 20
 find_devices() (in module ics), 52
 FindNeoDevices() (in module ics), 41
 fire (ics.DeviceSettings attribute), 25
 Fire2DeviceStatus (class in ics), 26
 fire2Status (ics.IcsDeviceStatus attribute), 28
 FireSettings (class in ics), 26
 firmware_update_required() (in module ics), 53
 FirmwareUpdateRequired() (in module ics), 41
 flags (ics.CmISO157652RxMessage attribute), 21
 flags (ics.CmISO157652TxMessage attribute), 22

flow_control (*ics.UartSettings* attribute), 37
 flowControlExtendedAddress
 (*ics.CmISO157652RxMessage* attribute),
 21
 flowControlExtendedAddress
 (*ics.CmISO157652TxMessage* attribute),
 22
 force_firmware_update() (in module *ics*), 53
 ForceFirmwareUpdate() (in module *ics*), 42
 fs_timeout (*ics.CmISO157652TxMessage* attribute),
 22
 fs_wait (*ics.CmISO157652TxMessage* attribute), 22

G

get_active_vnet_channel() (in module *ics*), 53
 get_backup_power_enabled() (in module *ics*),
 53
 get_backup_power_ready() (in module *ics*), 54
 get_bus_voltage() (in module *ics*), 54
 get_device_settings() (in module *ics*), 54
 get_device_status() (in module *ics*), 54
 get_dll_firmware_info() (in module *ics*), 55
 get_dll_version() (in module *ics*), 55
 get_error_messages() (in module *ics*), 55
 get_hw_firmware_info() (in module *ics*), 55
 get_last_api_error() (in module *ics*), 56
 get_library_path() (in module *ics*), 56
 get_messages() (in module *ics*), 56
 get_performance_parameters() (in module
 ics), 56
 get_rtc() (in module *ics*), 57
 get_script_status() (in module *ics*), 57
 get_serial_number() (in module *ics*), 57
 get_timestamp_for_msg() (in module *ics*), 57
 GetActiveVNETChannel() (in module *ics*), 42
 GetBackupPowerEnabled() (in module *ics*), 42
 GetBackupPowerReady() (in module *ics*), 42
 GetBusVoltage() (in module *ics*), 42
 GetDeviceStatus() (in module *ics*), 42
 GetDLLFirmwareInfo() (in module *ics*), 42
 GetDLLVersion() (in module *ics*), 42
 GetErrorMessages() (in module *ics*), 43
 GetFireSettings() (in module *ics*), 43
 GetHWFirmwareInfo() (in module *ics*), 43
 GetLastAPIError() (in module *ics*), 43
 GetMessages() (in module *ics*), 43
 GetPerformanceParameters() (in module *ics*),
 43
 GetRTC() (in module *ics*), 43
 GetSerialNumber() (in module *ics*), 43
 GetTimeStampForMsg() (in module *ics*), 43
 GetVCAN3Settings() (in module *ics*), 44
 GLOBAL_SETTINGS_SIZE (in module *ics*), 72
 GS_VERSION (in module *ics*), 72

H

Handle (*ics.NeoDevice* attribute), 29
 Header (*ics.SpyMessageJ1850* attribute), 35
 high_speed_auto_switch (*ics.SWCanSettings* at-
 tribute), 34
 hwComLatencyTestEn (*ics.RadStar2Settings* at-
 tribute), 32

I

iAppMajor (*ics.ApiFirmwareInfo* attribute), 20
 iAppMinor (*ics.ApiFirmwareInfo* attribute), 20
 iBoardRevMajor (*ics.ApiFirmwareInfo* attribute), 20
 iBoardRevMinor (*ics.ApiFirmwareInfo* attribute), 20
 iBootLoaderVersionMajor (*ics.ApiFirmwareInfo*
 attribute), 20
 iBootLoaderVersionMinor (*ics.ApiFirmwareInfo*
 attribute), 20
 ics (module), 19
 IcsDeviceStatus (class in *ics*), 28
 icsneoClosePort() (in module *ics*), 57
 icsneoEnableBusVoltageMonitor() (in mod-
 ule *ics*), 57
 icsneoEnableNetworkCom() (in module *ics*), 58
 icsneoFindNeoDevices() (in module *ics*), 58
 icsneoFirmwareUpdateRequired() (in module
 ics), 58
 icsneoForceFirmwareUpdate() (in module *ics*),
 58
 icsneoGetActiveVNETChannel() (in module
 ics), 58
 icsneoGetBackupPowerEnabled() (in module
 ics), 58
 icsneoGetBackupPowerReady() (in module *ics*),
 58
 icsneoGetBusVoltage() (in module *ics*), 58
 icsneoGetDeviceStatus() (in module *ics*), 59
 icsneoGetDLLFirmwareInfo() (in module *ics*),
 59
 icsneoGetDLLVersion() (in module *ics*), 59
 icsneoGetErrorMessages() (in module *ics*), 59
 icsneoGetFireSettings() (in module *ics*), 59
 icsneoGetHWFirmwareInfo() (in module *ics*), 59
 icsneoGetLastAPIError() (in module *ics*), 59
 icsneoGetMessages() (in module *ics*), 59
 icsneoGetPerformanceParameters() (in mod-
 ule *ics*), 59
 icsneoGetRTC() (in module *ics*), 60
 icsneoGetSerialNumber() (in module *ics*), 60
 icsneoGetTimeStampForMsg() (in module *ics*),
 60
 icsneoGetVCAN3Settings() (in module *ics*), 60
 icsneoISO15765_DisableNetworks() (in mod-
 ule *ics*), 60

icsneoISO15765_EnableNetworks() (in module ics), 60
 icsneoISO15765_ReceiveMessage() (in module ics), 60
 icsneoISO15765_TransmitMessage() (in module ics), 60
 icsneoLoadDefaultSettings() (in module ics), 61
 icsneoOpenNeoDevice() (in module ics), 61
 icsneoReadSDCard() (in module ics), 61
 icsneoRequestEnterSleepMode() (in module ics), 61
 icsneoScriptClear() (in module ics), 61
 icsneoScriptGetFBlockStatus() (in module ics), 61
 icsneoScriptGetScriptStatus() (in module ics), 61
 icsneoScriptGetScriptStatusEx() (in module ics), 61
 icsneoScriptLoad() (in module ics), 61
 icsneoScriptReadAppSignal() (in module ics), 62
 icsneoScriptReadRxMessage() (in module ics), 62
 icsneoScriptReadTxMessage() (in module ics), 62
 icsneoScriptStart() (in module ics), 62
 icsneoScriptStartFBlock() (in module ics), 62
 icsneoScriptStop() (in module ics), 62
 icsneoScriptStopFBlock() (in module ics), 62
 icsneoScriptWriteAppSignal() (in module ics), 62
 icsneoScriptWriteRxMessage() (in module ics), 63
 icsneoScriptWriteTxMessage() (in module ics), 63
 icsneoSetActiveVNETChannel() (in module ics), 63
 icsneoSetBackupPowerEnabled() (in module ics), 63
 icsneoSetBitRate() (in module ics), 63
 icsneoSetBitRateEx() (in module ics), 63
 icsneoSetContext() (in module ics), 63
 icsneoSetFDBitRate() (in module ics), 63
 icsneoSetFireSettings() (in module ics), 64
 icsneoSetReflashDisplayCallbacks() (in module ics), 64
 icsneoSetRTC() (in module ics), 64
 icsneoSetVCAN3Settings() (in module ics), 64
 icsneoTxMessages() (in module ics), 64
 icsneoValidateHObject() (in module ics), 64
 icsneoWriteSDCard() (in module ics), 64
 id (ics.CmISO157652RxMessage attribute), 21
 id (ics.CmISO157652TxMessage attribute), 22
 id_mask (ics.CmISO157652RxMessage attribute), 21
 idle_wakeup_network_enables_1 (ics.RadGalaxySettings attribute), 31
 idle_wakeup_network_enables_1 (ics.RadStar2Settings attribute), 32
 idle_wakeup_network_enables_1 (ics.VcanRFSettings attribute), 40
 idle_wakeup_network_enables_2 (ics.RadGalaxySettings attribute), 31
 idle_wakeup_network_enables_2 (ics.RadStar2Settings attribute), 32
 idle_wakeup_network_enables_2 (ics.VcanRFSettings attribute), 40
 idle_wakeup_network_enables_3 (ics.CyanSettings attribute), 24
 idle_wakeup_network_enables_3 (ics.RadGalaxySettings attribute), 31
 idle_wakeup_network_enables_3 (ics.RadStar2Settings attribute), 32
 iMainFirmChkSum (ics.ApiFirmwareInfo attribute), 20
 iMainFirmDateDay (ics.ApiFirmwareInfo attribute), 20
 iMainFirmDateHour (ics.ApiFirmwareInfo attribute), 20
 iMainFirmDateMin (ics.ApiFirmwareInfo attribute), 20
 iMainFirmDateMonth (ics.ApiFirmwareInfo attribute), 20
 iMainFirmDateSecond (ics.ApiFirmwareInfo attribute), 20
 iMainFirmDateYear (ics.ApiFirmwareInfo attribute), 20
 iMainVnetHWrevMajor (ics.ApiFirmwareInfo attribute), 20
 iMainVnetHWrevMinor (ics.ApiFirmwareInfo attribute), 20
 iMainVnetSRAMSize (ics.ApiFirmwareInfo attribute), 20
 iManufactureDay (ics.ApiFirmwareInfo attribute), 20
 iManufactureMonth (ics.ApiFirmwareInfo attribute), 20
 iManufactureYear (ics.ApiFirmwareInfo attribute), 20
 init_steps (ics.Iso9141Keyword2000Settings attribute), 28
 innerFrameDelay25us (ics.CanSettings attribute), 21
 ISO15765_2_NETWORK_HSCAN (in module ics), 72
 ISO15765_2_NETWORK_HSCAN2 (in module ics), 72
 ISO15765_2_NETWORK_HSCAN3 (in module ics), 72
 ISO15765_2_NETWORK_HSCAN4 (in module ics), 72
 ISO15765_2_NETWORK_HSCAN5 (in module ics), 72

ISO15765_2_NETWORK_HSCAN6 (in module ics), 72
 ISO15765_2_NETWORK_HSCAN7 (in module ics), 72
 ISO15765_2_NETWORK_MSCAN (in module ics), 72
 ISO15765_2_NETWORK_SWCAN (in module ics), 72
 ISO15765_2_NETWORK_SWCAN2 (in module ics), 72
 iso15765_disable_networks() (in module ics), 64
 ISO15765_DisableNetworks() (in module ics), 44
 iso15765_enable_networks() (in module ics), 64
 ISO15765_EnableNetworks() (in module ics), 44
 iso15765_receive_message() (in module ics), 65
 ISO15765_ReceiveMessage() (in module ics), 44
 iso15765_separation_time_offset (ics.CyanSettings attribute), 24
 iso15765_separation_time_offset (ics.FireSettings attribute), 26
 iso15765_separation_time_offset (ics.RadGalaxySettings attribute), 31
 iso15765_separation_time_offset (ics.RadStar2Settings attribute), 32
 iso15765_separation_time_offset (ics.Vcan3Settings attribute), 37
 iso15765_separation_time_offset (ics.Vcan4I2Settings attribute), 38
 iso15765_separation_time_offset (ics.Vcan4Settings attribute), 39
 iso15765_separation_time_offset (ics.VcanRFSettings attribute), 40
 iso15765_separation_time_offset (ics.VividCANSettings attribute), 41
 iso15765_transmit_message() (in module ics), 65
 ISO15765_TransmitMessage() (in module ics), 44
 ISO9141_KEYWORD2000__INIT_STEP_SIZE (in module ics), 72
 ISO9141_KEYWORD2000_SETTINGS_SIZE (in module ics), 72
 iso9141_kwp_enable_reserved (ics.FireSettings attribute), 27
 iso9141_kwp_enable_reserved (ics.VcanRFSettings attribute), 40
 iso9141_kwp_settings (ics.FireSettings attribute), 27
 iso9141_kwp_settings (ics.VcanRFSettings attribute), 40
 iso9141_kwp_settings_1 (ics.CyanSettings attribute), 24
 iso9141_kwp_settings_1 (ics.RadGalaxySettings attribute), 31
 iso9141_kwp_settings_1 (ics.RadStar2Settings attribute), 32
 iso9141_kwp_settings_1 (ics.Vcan4Settings attribute), 39
 iso9141_kwp_settings_2 (ics.CyanSettings attribute), 24
 iso9141_kwp_settings_2 (ics.FireSettings attribute), 27
 iso9141_kwp_settings_2 (ics.VcanRFSettings attribute), 40
 iso9141_kwp_settings_3 (ics.CyanSettings attribute), 24
 iso9141_kwp_settings_3 (ics.FireSettings attribute), 27
 iso9141_kwp_settings_4 (ics.CyanSettings attribute), 24
 iso9141_kwp_settings_4 (ics.FireSettings attribute), 27
 Iso9141Keyword2000InitSteps (class in ics), 28
 Iso9141Keyword2000Settings (class in ics), 28
 iso_9141_kwp_enable_reserved (ics.RadStar2Settings attribute), 32
 iso_9141_kwp_enable_reserved (ics.Vcan4Settings attribute), 39
 iso_msg_termination (ics.FireSettings attribute), 27
 iso_msg_termination (ics.VcanRFSettings attribute), 40
 iso_msg_termination_1 (ics.CyanSettings attribute), 24
 iso_msg_termination_1 (ics.RadGalaxySettings attribute), 31
 iso_msg_termination_1 (ics.RadStar2Settings attribute), 32
 iso_msg_termination_1 (ics.Vcan4Settings attribute), 39
 iso_msg_termination_2 (ics.CyanSettings attribute), 24
 iso_msg_termination_2 (ics.FireSettings attribute), 27
 iso_msg_termination_2 (ics.VcanRFSettings attribute), 40
 iso_msg_termination_3 (ics.CyanSettings attribute), 24
 iso_msg_termination_3 (ics.FireSettings attribute), 27
 iso_msg_termination_4 (ics.CyanSettings attribute), 24
 iso_msg_termination_4 (ics.FireSettings attribute), 27
 iso_parity (ics.FireSettings attribute), 27
 iso_parity (ics.VcanRFSettings attribute), 40
 iso_parity_1 (ics.CyanSettings attribute), 24
 iso_parity_1 (ics.RadGalaxySettings attribute), 31
 iso_parity_1 (ics.RadStar2Settings attribute), 32

iso_parity_1 (*ics.Vcan4Settings* attribute), 39
 iso_parity_2 (*ics.CyanSettings* attribute), 24
 iso_parity_2 (*ics.FireSettings* attribute), 27
 iso_parity_2 (*ics.VcanRFSettings* attribute), 40
 iso_parity_3 (*ics.CyanSettings* attribute), 24
 iso_parity_3 (*ics.FireSettings* attribute), 27
 iso_parity_4 (*ics.CyanSettings* attribute), 24
 iso_parity_4 (*ics.FireSettings* attribute), 27
 iso_tester_pullup_enable (*ics.FireSettings* attribute), 27
 iso_tester_pullup_enable (*ics.VcanRFSettings* attribute), 40
 IsOpen (*ics.NeoDevice* attribute), 29
 iType (*ics.ApiFirmwareInfo* attribute), 20

J

J1708_SETTINGS_SIZE (in module *ics*), 72

K

k (*ics.Iso9141Keyword2000InitSteps* attribute), 28

L

l (*ics.Iso9141Keyword2000InitSteps* attribute), 28
 led_mode (*ics.EthernetSettings* attribute), 26
 lin1 (*ics.CyanSettings* attribute), 24
 lin1 (*ics.FireSettings* attribute), 27
 lin1 (*ics.RadGalaxySettings* attribute), 31
 lin1 (*ics.RadStar2Settings* attribute), 33
 lin1 (*ics.Vcan4Settings* attribute), 39
 lin1 (*ics.VcanRFSettings* attribute), 40
 lin2 (*ics.CyanSettings* attribute), 24
 lin2 (*ics.FireSettings* attribute), 27
 lin2 (*ics.VcanRFSettings* attribute), 40
 lin3 (*ics.CyanSettings* attribute), 24
 lin3 (*ics.FireSettings* attribute), 27
 lin4 (*ics.CyanSettings* attribute), 24
 lin4 (*ics.FireSettings* attribute), 27
 lin5 (*ics.CyanSettings* attribute), 24
 lin6 (*ics.CyanSettings* attribute), 24
 LIN_SETTINGS_SIZE (in module *ics*), 72
 link_speed (*ics.EthernetSettings* attribute), 26
 LinSettings (class in *ics*), 29
 LISTEN_ALL (in module *ics*), 72
 LISTEN_ONLY (in module *ics*), 72
 load_default_settings() (in module *ics*), 65
 LoadDefaultSettings() (in module *ics*), 44
 LOOPBACK (in module *ics*), 72
 lsft (*ics.FireSettings* attribute), 27
 lsft1 (*ics.CyanSettings* attribute), 24
 lsft2 (*ics.CyanSettings* attribute), 24
 lsftcan1 (*ics.VividCANSSettings* attribute), 41

M

MasterEnable (*ics.TimesyncSettings* attribute), 37

MasterNetwork (*ics.TimesyncSettings* attribute), 37
 MasterResistor (*ics.LinSettings* attribute), 29
 MaxAllowedClients (*ics.NeoDevice* attribute), 29
 MessagePieceID (*ics.SpyMessage* attribute), 34
 MessagePieceID (*ics.SpyMessageJ1850* attribute), 35
 misc_io_analog_enable (*ics.CyanSettings* attribute), 25
 misc_io_analog_enable (*ics.FireSettings* attribute), 27
 misc_io_analog_enable (*ics.RadGalaxySettings* attribute), 31
 misc_io_analog_enable (*ics.RadStar2Settings* attribute), 33
 misc_io_analog_enable (*ics.VcanRFSettings* attribute), 40
 misc_io_initial_ddr (*ics.CyanSettings* attribute), 25
 misc_io_initial_ddr (*ics.FireSettings* attribute), 27
 misc_io_initial_ddr (*ics.RadGalaxySettings* attribute), 31
 misc_io_initial_ddr (*ics.RadStar2Settings* attribute), 33
 misc_io_initial_ddr (*ics.Vcan3Settings* attribute), 37
 misc_io_initial_ddr (*ics.VcanRFSettings* attribute), 40
 misc_io_initial_latch (*ics.CyanSettings* attribute), 25
 misc_io_initial_latch (*ics.FireSettings* attribute), 27
 misc_io_initial_latch (*ics.RadGalaxySettings* attribute), 31
 misc_io_initial_latch (*ics.RadStar2Settings* attribute), 33
 misc_io_initial_latch (*ics.Vcan3Settings* attribute), 37
 misc_io_initial_latch (*ics.VcanRFSettings* attribute), 40
 misc_io_on_report_events (*ics.CyanSettings* attribute), 25
 misc_io_on_report_events (*ics.FireSettings* attribute), 27
 misc_io_on_report_events (*ics.RadGalaxySettings* attribute), 31
 misc_io_on_report_events (*ics.RadStar2Settings* attribute), 33
 misc_io_on_report_events (*ics.Vcan3Settings* attribute), 37
 misc_io_on_report_events (*ics.VcanRFSettings* attribute), 40
 misc_io_report_period (*ics.CyanSettings* attribute), 25

`misc_io_report_period` (*ics.FireSettings attribute*), 27
`misc_io_report_period` (*ics.RadGalaxySettings attribute*), 31
`misc_io_report_period` (*ics.RadStar2Settings attribute*), 33
`misc_io_report_period` (*ics.Vcan3Settings attribute*), 37
`misc_io_report_period` (*ics.VcanRFSettings attribute*), 40
`MiscData` (*ics.SpyMessage attribute*), 34
`MiscData` (*ics.SpyMessageJ1850 attribute*), 35
`Mode` (*ics.CanSettings attribute*), 21
`Mode` (*ics.LinSettings attribute*), 29
`Mode` (*ics.SWCanSettings attribute*), 33

N

`Name` (*ics.NeoDevice attribute*), 29
`NEO_CFG_MPIC_HS_CAN_CNF1` (*in module ics*), 74
`NEO_CFG_MPIC_HS_CAN_CNF2` (*in module ics*), 74
`NEO_CFG_MPIC_HS_CAN_CNF3` (*in module ics*), 74
`NEO_CFG_MPIC_HS_CAN_MODE` (*in module ics*), 74
`NEO_CFG_MPIC_LSFT_CAN_CNF1` (*in module ics*), 74
`NEO_CFG_MPIC_LSFT_CAN_CNF2` (*in module ics*), 74
`NEO_CFG_MPIC_LSFT_CAN_CNF3` (*in module ics*), 74
`NEO_CFG_MPIC_MS_CAN_CNF1` (*in module ics*), 74
`NEO_CFG_MPIC_MS_CAN_CNF2` (*in module ics*), 74
`NEO_CFG_MPIC_MS_CAN_CNF3` (*in module ics*), 74
`NEO_CFG_MPIC_SW_CAN_CNF1` (*in module ics*), 74
`NEO_CFG_MPIC_SW_CAN_CNF2` (*in module ics*), 74
`NEO_CFG_MPIC_SW_CAN_CNF3` (*in module ics*), 74
`NeoDevice` (*class in ics*), 29
`NEODEVICE_ALL` (*in module ics*), 72
`NEODEVICE_ANY_ION` (*in module ics*), 73
`NEODEVICE_ANY_PLASMA` (*in module ics*), 73
`NEODEVICE_BLUE` (*in module ics*), 73
`NEODEVICE_CMPROBE` (*in module ics*), 73
`NEODEVICE_CT_OBD` (*in module ics*), 73
`NEODEVICE_DW_VCAN` (*in module ics*), 73
`NEODEVICE_ECU` (*in module ics*), 73
`NEODEVICE_ECUCHIP_UART` (*in module ics*), 73
`NEODEVICE_EEVB` (*in module ics*), 73
`NEODEVICE_FIRE` (*in module ics*), 73
`NEODEVICE_FIRE2` (*in module ics*), 73
`NEODEVICE_FIRE_VNET` (*in module ics*), 73
`NEODEVICE_FLEX` (*in module ics*), 73
`NEODEVICE_IEVB` (*in module ics*), 73
`NEODEVICE_ION_2` (*in module ics*), 73
`NEODEVICE_ION_3` (*in module ics*), 73
`NEODEVICE_NEOANALOG` (*in module ics*), 73
`NEODEVICE_NEOECUCHIP` (*in module ics*), 73

`NEODEVICE_OBD2_PRO` (*in module ics*), 73
`NEODEVICE_OBD2_SIM` (*in module ics*), 73
`NEODEVICE_PENDANT` (*in module ics*), 73
`NEODEVICE_PLASMA_1_11` (*in module ics*), 73
`NEODEVICE_PLASMA_1_12` (*in module ics*), 73
`NEODEVICE_PLASMA_1_13` (*in module ics*), 73
`NEODEVICE_RADGALAXY` (*in module ics*), 73
`NEODEVICE_RADSTAR` (*in module ics*), 73
`NEODEVICE_RADSTAR2` (*in module ics*), 73
`NEODEVICE_RED` (*in module ics*), 73
`NEODEVICE_SW_VCAN` (*in module ics*), 73
`NEODEVICE_UNKNOWN` (*in module ics*), 73
`NEODEVICE_VCAN3` (*in module ics*), 73
`NEODEVICE_VCAN4` (*in module ics*), 73
`NEODEVICE_VCAN4_12` (*in module ics*), 73
`NEODEVICE_VCANRF` (*in module ics*), 73
`NEODEVICE_VIVIDCAN` (*in module ics*), 73
`NEOVI6_VCAN_TIMESTAMP_1` (*in module ics*), 73
`NEOVI6_VCAN_TIMESTAMP_2` (*in module ics*), 74
`NEOVI_3G_MAX_SETTINGS_SIZE` (*in module ics*), 74
`NEOVI_COMMTYPE_FIRE_USB` (*in module ics*), 74
`NEOVI_COMMTYPE_RS232` (*in module ics*), 74
`NEOVI_COMMTYPE_TCPIP` (*in module ics*), 74
`NEOVI_COMMTYPE_USB_BULK` (*in module ics*), 74
`NEOVI_RED_TIMESTAMP_1_10NS` (*in module ics*), 74
`NEOVI_RED_TIMESTAMP_1_25NS` (*in module ics*), 74
`NEOVI_RED_TIMESTAMP_2_10NS` (*in module ics*), 74
`NEOVI_RED_TIMESTAMP_2_25NS` (*in module ics*), 74
`NEOVI_TIMESTAMP_1` (*in module ics*), 74
`NEOVI_TIMESTAMP_2` (*in module ics*), 74
`NEOVI6_VCAN_TIMESTAMP_1` (*in module ics*), 74
`NEOVI6_VCAN_TIMESTAMP_2` (*in module ics*), 74
`NETID_3G_APP_SIGNAL_STATUS` (*in module ics*), 74
`NETID_3G_FB_STATUS` (*in module ics*), 74
`NETID_3G_LOGGING_OVERFLOW` (*in module ics*), 74
`NETID_3G_READ_DATA LINK_CM_RX_MSG` (*in module ics*), 74
`NETID_3G_READ_DATA LINK_CM_TX_MSG` (*in module ics*), 74
`NETID_3G_READ_SETTINGS_EX` (*in module ics*), 74
`NETID_3G_RESET_STATUS` (*in module ics*), 74
`NETID_AUX` (*in module ics*), 74
`NETID_CGI` (*in module ics*), 74
`NETID_DATA_TO_HOST` (*in module ics*), 75
`NETID_DEVICE` (*in module ics*), 75
`NETID_DEVICE_STATUS` (*in module ics*), 75
`NETID_ETHERNET` (*in module ics*), 75
`NETID_ETHERNET_DAQ` (*in module ics*), 75

- NETID_FLEXRAY (in module ics), 75
- NETID_FLEXRAY1A (in module ics), 75
- NETID_FLEXRAY1B (in module ics), 75
- NETID_FLEXRAY2 (in module ics), 75
- NETID_FLEXRAY2A (in module ics), 75
- NETID_FLEXRAY2B (in module ics), 75
- NETID_FORDSCP (in module ics), 75
- NETID_GMFSA (in module ics), 75
- NETID_HSCAN (in module ics), 75
- NETID_HSCAN2 (in module ics), 75
- NETID_HSCAN3 (in module ics), 75
- NETID_HSCAN4 (in module ics), 75
- NETID_HSCAN5 (in module ics), 75
- NETID_HSCAN6 (in module ics), 75
- NETID_HSCAN7 (in module ics), 75
- NETID_HW_COM_LATENCY_TEST (in module ics), 75
- NETID_I2C1 (in module ics), 75
- NETID_INVALID (in module ics), 75
- NETID_ISO (in module ics), 75
- NETID_ISO14230 (in module ics), 75
- NETID_ISO2 (in module ics), 75
- NETID_ISO3 (in module ics), 75
- NETID_ISO4 (in module ics), 75
- NETID_ISOPIC (in module ics), 75
- NETID_J1708 (in module ics), 75
- NETID_JVPW (in module ics), 75
- NETID_LIN (in module ics), 75
- NETID_LIN2 (in module ics), 75
- NETID_LIN3 (in module ics), 75
- NETID_LIN4 (in module ics), 75
- NETID_LIN5 (in module ics), 75
- NETID_LIN6 (in module ics), 76
- NETID_LSFTCAN (in module ics), 76
- NETID_LSFTCAN2 (in module ics), 76
- NETID_MAIN51 (in module ics), 76
- NETID_MAX (in module ics), 76
- NETID_MOST (in module ics), 76
- NETID_MOST150 (in module ics), 76
- NETID_MOST25 (in module ics), 76
- NETID_MOST50 (in module ics), 76
- NETID_MSCAN (in module ics), 76
- NETID_OP_ETHERNET1 (in module ics), 76
- NETID_OP_ETHERNET10 (in module ics), 76
- NETID_OP_ETHERNET11 (in module ics), 76
- NETID_OP_ETHERNET12 (in module ics), 76
- NETID_OP_ETHERNET2 (in module ics), 76
- NETID_OP_ETHERNET3 (in module ics), 76
- NETID_OP_ETHERNET4 (in module ics), 76
- NETID_OP_ETHERNET5 (in module ics), 76
- NETID_OP_ETHERNET6 (in module ics), 76
- NETID_OP_ETHERNET7 (in module ics), 76
- NETID_OP_ETHERNET8 (in module ics), 76
- NETID_OP_ETHERNET9 (in module ics), 76
- NETID_RED (in module ics), 76
- NETID_RED_APP_ERROR (in module ics), 76
- NETID_RED_VBAT (in module ics), 76
- NETID_RS232 (in module ics), 76
- NETID_SCI (in module ics), 76
- NETID_SPI1 (in module ics), 76
- NETID_SWCAN (in module ics), 76
- NETID_SWCAN2 (in module ics), 76
- NETID_TCP (in module ics), 76
- NETID_TEXTAPI_TO_HOST (in module ics), 76
- NETID_UART (in module ics), 76
- NETID_UART2 (in module ics), 76
- NETID_UART3 (in module ics), 76
- NETID_UART4 (in module ics), 76
- network_enabled_on_boot (ics.CyanSettings attribute), 25
- network_enabled_on_boot (ics.FireSettings attribute), 27
- network_enabled_on_boot (ics.RadGalaxySettings attribute), 31
- network_enabled_on_boot (ics.RadStar2Settings attribute), 33
- network_enabled_on_boot (ics.Vcan3Settings attribute), 37
- network_enabled_on_boot (ics.Vcan412Settings attribute), 38
- network_enabled_on_boot (ics.Vcan4Settings attribute), 39
- network_enabled_on_boot (ics.VcanRFSettings attribute), 40
- network_enabled_on_boot (ics.VividCANSettings attribute), 41
- network_enables (ics.CyanSettings attribute), 25
- network_enables (ics.FireSettings attribute), 28
- network_enables (ics.RadGalaxySettings attribute), 31
- network_enables (ics.RadStar2Settings attribute), 33
- network_enables (ics.TextApiSettings attribute), 36
- network_enables (ics.Vcan3Settings attribute), 37
- network_enables (ics.Vcan412Settings attribute), 38
- network_enables (ics.Vcan4Settings attribute), 39
- network_enables (ics.VcanRFSettings attribute), 40
- network_enables (ics.VividCANSettings attribute), 41
- network_enables_2 (ics.CyanSettings attribute), 25
- network_enables_2 (ics.FireSettings attribute), 28
- network_enables_2 (ics.RadGalaxySettings attribute), 31
- network_enables_2 (ics.RadStar2Settings attribute), 33
- network_enables_2 (ics.Vcan4Settings attribute), 39
- network_enables_2 (ics.VcanRFSettings attribute), 39

40
network_enables_3 (*ics.CyanSettings* attribute), 25
network_enables_3 (*ics.RadGalaxySettings* attribute), 31
network_enables_3 (*ics.RadStar2Settings* attribute), 33
network_enables_3 (*ics.Vcan4Settings* attribute), 39
NetworkID (*ics.SpyMessage* attribute), 34
NetworkID (*ics.SpyMessageJ1850* attribute), 35
NetworkID2 (*ics.SpyMessage* attribute), 34
NetworkID2 (*ics.SpyMessageJ1850* attribute), 35
NO_CANFD (*in module ics*), 77
NodeID (*ics.SpyMessage* attribute), 34
NodeID (*ics.SpyMessageJ1850* attribute), 35
noExtraDataPtrCleanup (*ics.SpyMessage* attribute), 35
noExtraDataPtrCleanup (*ics.SpyMessageJ1850* attribute), 36
NORMAL (*in module ics*), 77
NORMAL_MODE (*in module ics*), 77
num_bytes (*ics.CmISO157652TxMessage* attribute), 22
NumberBytesData (*ics.SpyMessage* attribute), 34
NumberBytesData (*ics.SpyMessageJ1850* attribute), 35
NumberBytesHeader (*ics.SpyMessage* attribute), 34
NumberBytesHeader (*ics.SpyMessageJ1850* attribute), 35
NumberOfClients (*ics.NeoDevice* attribute), 29

O

OP_ETH_GENERAL_SETTINGS_SIZE (*in module ics*), 77
OP_ETH_SETTINGS_SIZE (*in module ics*), 77
open_device() (*in module ics*), 65
OpenNeoDevice() (*in module ics*), 44
opEth1 (*ics.RadGalaxySettings* attribute), 31
opEth1 (*ics.RadStar2Settings* attribute), 33
opEth10 (*ics.RadGalaxySettings* attribute), 31
opEth11 (*ics.RadGalaxySettings* attribute), 31
opEth12 (*ics.RadGalaxySettings* attribute), 31
opEth2 (*ics.RadGalaxySettings* attribute), 31
opEth2 (*ics.RadStar2Settings* attribute), 33
opEth3 (*ics.RadGalaxySettings* attribute), 31
opEth4 (*ics.RadGalaxySettings* attribute), 31
opEth5 (*ics.RadGalaxySettings* attribute), 31
opEth6 (*ics.RadGalaxySettings* attribute), 31
opEth7 (*ics.RadGalaxySettings* attribute), 31
opEth8 (*ics.RadGalaxySettings* attribute), 32
opEth9 (*ics.RadGalaxySettings* attribute), 32
OPETH_FUNC_MEDIACONVERTER (*in module ics*), 77
OPETH_FUNC_TAP (*in module ics*), 77

OPETH_FUNC_TAP_LOW_LATENCY (*in module ics*), 77
OPETH_LINK_AUTO (*in module ics*), 77
OPETH_LINK_MASTER (*in module ics*), 77
OPETH_LINK_SLAVE (*in module ics*), 77
OPETH_MAC_SPOOF_DST_ADDR (*in module ics*), 77
OPETH_MAC_SPOOF_SRC_ADDR (*in module ics*), 77
opEthGen (*ics.RadGalaxySettings* attribute), 32
opEthGen (*ics.RadStar2Settings* attribute), 33
OpEthGeneralSettings (*class in ics*), 29
OpEthSettings (*class in ics*), 29
override_library_name() (*in module ics*), 66

P

p2_500us (*ics.Iso9141Keyword2000Settings* attribute), 28
p3_500us (*ics.Iso9141Keyword2000Settings* attribute), 28
p4_500us (*ics.Iso9141Keyword2000Settings* attribute), 28
padding (*ics.CmISO157652RxMessage* attribute), 22
padding (*ics.CmISO157652TxMessage* attribute), 22
parity (*ics.UartSettings* attribute), 37
pc_com_mode (*ics.RadStar2Settings* attribute), 33
perf_en (*ics.CyanSettings* attribute), 25
perf_en (*ics.FireSettings* attribute), 28
perf_en (*ics.RadGalaxySettings* attribute), 32
perf_en (*ics.RadStar2Settings* attribute), 33
perf_en (*ics.Vcan3Settings* attribute), 37
perf_en (*ics.Vcan412Settings* attribute), 38
perf_en (*ics.Vcan4Settings* attribute), 39
perf_en (*ics.VcanRFSettings* attribute), 40
perf_en (*ics.VividCANSSettings* attribute), 41
PLASMA_SLAVE1_OFFSET (*in module ics*), 77
PLASMA_SLAVE1_OFFSET_RANGE2 (*in module ics*), 77
PLASMA_SLAVE2_OFFSET (*in module ics*), 77
PLASMA_SLAVE2_OFFSET_RANGE2 (*in module ics*), 77
PLASMA_SLAVE3_OFFSET_RANGE2 (*in module ics*), 77
PLASMA_SLAVE_NUM (*in module ics*), 77
preemption_en (*ics.OpEthSettings* attribute), 30
Protocol (*ics.SpyMessage* attribute), 34
Protocol (*ics.SpyMessageJ1850* attribute), 35
pwm_man_timeout (*ics.FireSettings* attribute), 28
pwr_man_enable (*ics.CyanSettings* attribute), 25
pwr_man_enable (*ics.FireSettings* attribute), 28
pwr_man_enable (*ics.RadGalaxySettings* attribute), 32
pwr_man_enable (*ics.RadStar2Settings* attribute), 33
pwr_man_enable (*ics.Vcan412Settings* attribute), 38
pwr_man_enable (*ics.Vcan4Settings* attribute), 39
pwr_man_enable (*ics.VcanRFSettings* attribute), 40

pwr_man_enable (*ics.VividCANSettings attribute*), 41
 pwr_man_timeout (*ics.CyanSettings attribute*), 25
 pwr_man_timeout (*ics.RadGalaxySettings attribute*), 32
 pwr_man_timeout (*ics.RadStar2Settings attribute*), 33
 pwr_man_timeout (*ics.Vcan412Settings attribute*), 38
 pwr_man_timeout (*ics.Vcan4Settings attribute*), 39
 pwr_man_timeout (*ics.VcanRFSettings attribute*), 40
 pwr_man_timeout (*ics.VividCANSettings attribute*), 41

R

radgalaxy (*ics.DeviceSettings attribute*), 25
 RadGalaxySettings (*class in ics*), 30
 radstar2 (*ics.DeviceSettings attribute*), 25
 RadStar2Settings (*class in ics*), 32
 read_sdcard() (*in module ics*), 66
 ReadSDCard() (*in module ics*), 44
 REPORT_ON_GPS (*in module ics*), 77
 REPORT_ON_KLINE (*in module ics*), 77
 REPORT_ON_LED1 (*in module ics*), 77
 REPORT_ON_LED2 (*in module ics*), 77
 REPORT_ON_MISC1 (*in module ics*), 77
 REPORT_ON_MISC2 (*in module ics*), 77
 REPORT_ON_MISC3 (*in module ics*), 77
 REPORT_ON_MISC3_AIN (*in module ics*), 77
 REPORT_ON_MISC4 (*in module ics*), 77
 REPORT_ON_MISC4_AIN (*in module ics*), 77
 REPORT_ON_MISC5 (*in module ics*), 77
 REPORT_ON_MISC5_AIN (*in module ics*), 77
 REPORT_ON_MISC6 (*in module ics*), 77
 REPORT_ON_MISC6_AIN (*in module ics*), 77
 REPORT_ON_PERIODIC (*in module ics*), 77
 REPORT_ON_PWM_IN1 (*in module ics*), 77
 request_enter_sleep_mode() (*in module ics*), 66
 RequestEnterSleepMode() (*in module ics*), 45
 reserved (*ics.CmISO157652RxMessage attribute*), 22
 reserved (*ics.CyanSettings attribute*), 25
 RESERVED (*ics.SWCanSettings attribute*), 33
 reserved (*ics.Vcan412Settings attribute*), 38
 reserved (*ics.Vcan4Settings attribute*), 39
 reserved (*ics.VividCANSettings attribute*), 41
 reserved0 (*ics.OpEthGeneralSettings attribute*), 29
 reserved0 (*ics.OpEthSettings attribute*), 30
 reserved_1 (*ics.UartSettings attribute*), 37
 RESISTOR_OFF (*in module ics*), 77
 RESISTOR_ON (*in module ics*), 78
 rsvd (*ics.EthernetSettings attribute*), 26
 RuntimeError, 20

S

SCRIPT_LOCATION_FLASH_MEM (*in module ics*), 78
 SCRIPT_LOCATION_INTERNAL_FLASH (*in module ics*), 78
 SCRIPT_LOCATION_SDCARD (*in module ics*), 78
 SCRIPT_LOCATION_VCAN3_MEM (*in module ics*), 78
 SCRIPT_STATUS_RUNNING (*in module ics*), 78
 SCRIPT_STATUS_STOPPED (*in module ics*), 78
 ScriptClear() (*in module ics*), 45
 ScriptGetFBlockStatus() (*in module ics*), 45
 ScriptGetScriptStatus() (*in module ics*), 45
 ScriptGetScriptStatusEx() (*in module ics*), 45
 ScriptLoad() (*in module ics*), 45
 ScriptReadAppSignal() (*in module ics*), 45
 ScriptReadRxMessage() (*in module ics*), 45
 ScriptReadTxMessage() (*in module ics*), 46
 ScriptStart() (*in module ics*), 46
 ScriptStartFBlock() (*in module ics*), 46
 ScriptStop() (*in module ics*), 46
 ScriptStopFBlock() (*in module ics*), 46
 ScriptWriteAppSignal() (*in module ics*), 46
 ScriptWriteRxMessage() (*in module ics*), 46
 ScriptWriteTxMessage() (*in module ics*), 46
 SerialNumber (*ics.NeoDevice attribute*), 29
 set_active_vnet_channel() (*in module ics*), 66
 set_backup_power_enabled() (*in module ics*), 67
 set_bit_rate() (*in module ics*), 67
 set_bit_rate_ex() (*in module ics*), 67
 set_context() (*in module ics*), 67
 set_device_settings() (*in module ics*), 67
 set_fd_bit_rate() (*in module ics*), 68
 set_reflash_callback() (*in module ics*), 68
 set_rtc() (*in module ics*), 68
 SetActiveVNETChannel() (*in module ics*), 46
 SetBackupPowerEnabled() (*in module ics*), 47
 SetBaudrate (*ics.CanSettings attribute*), 21
 SetBaudrate (*ics.SWCanSettings attribute*), 33
 SetBitRate() (*in module ics*), 47
 SetBitRateEx() (*in module ics*), 47
 SetContext() (*in module ics*), 47
 SetFDBitRate() (*in module ics*), 47
 SetFireSettings() (*in module ics*), 47
 SetReflashDisplayCallback() (*in module ics*), 47
 SetRTC() (*in module ics*), 47
 SetVCAN3Settings() (*in module ics*), 48
 SlaveEnable (*ics.TimesyncSettings attribute*), 37
 SlaveNetwork (*ics.TimesyncSettings attribute*), 37
 slaveVnetA (*ics.CyanSettings attribute*), 25
 slaveVnetB (*ics.CyanSettings attribute*), 25
 SLEEP_MODE (*in module ics*), 78
 SLOW_MODE (*in module ics*), 78
 spbrg (*ics.Iso9141Keyword2000Settings attribute*), 28

spbrg (*ics.LinSettings attribute*), 29
spbrg (*ics.UartSettings attribute*), 37
SPY_PROTOCOL_BEAN (*in module ics*), 78
SPY_PROTOCOL_CAN (*in module ics*), 78
SPY_PROTOCOL_CANFD (*in module ics*), 78
SPY_PROTOCOL_CGI (*in module ics*), 78
SPY_PROTOCOL_CHRYSLER_CCD (*in module ics*), 78
SPY_PROTOCOL_CHRYSLER_JVPW (*in module ics*), 78
SPY_PROTOCOL_CHRYSLER_SCI (*in module ics*), 78
SPY_PROTOCOL_CUSTOM (*in module ics*), 78
SPY_PROTOCOL_DALLAS_1WIRE (*in module ics*), 78
SPY_PROTOCOL_ETHERNET (*in module ics*), 78
SPY_PROTOCOL_FLEXRAY (*in module ics*), 78
SPY_PROTOCOL_FORD_UBP (*in module ics*), 78
SPY_PROTOCOL_GENERIC_MANCHESTER (*in module ics*), 78
SPY_PROTOCOL_GENERIC_UART (*in module ics*), 78
SPY_PROTOCOL_GM_ALDL_UART (*in module ics*), 78
SPY_PROTOCOL_GME_CIM_SCL_KLINE (*in module ics*), 78
SPY_PROTOCOL_GMFSA (*in module ics*), 78
SPY_PROTOCOL_GMLAN (*in module ics*), 78
SPY_PROTOCOL_I2C (*in module ics*), 78
SPY_PROTOCOL_ISO9141 (*in module ics*), 78
SPY_PROTOCOL_J1708 (*in module ics*), 78
SPY_PROTOCOL_J1850PWM (*in module ics*), 78
SPY_PROTOCOL_J1850VPW (*in module ics*), 78
SPY_PROTOCOL_J1939 (*in module ics*), 78
SPY_PROTOCOL_JTAG (*in module ics*), 78
SPY_PROTOCOL_LIN (*in module ics*), 78
SPY_PROTOCOL_MOST (*in module ics*), 78
SPY_PROTOCOL_SENT_PROTOCOL (*in module ics*), 79
SPY_PROTOCOL_SPI (*in module ics*), 79
SPY_PROTOCOL_TCP (*in module ics*), 79
SPY_PROTOCOL_UART (*in module ics*), 79
SPY_PROTOCOL_UNIO (*in module ics*), 79
SPY_STATUS2_CAN_HAVE_LINK_DATA (*in module ics*), 79
SPY_STATUS2_CAN_ISO15765_LOGICAL_FRAME (*in module ics*), 79
SPY_STATUS2_END_OF_LONG_MESSAGE (*in module ics*), 79
SPY_STATUS2_ERROR_FRAME (*in module ics*), 79
SPY_STATUS2_ETHERNET_CRC_ERROR (*in module ics*), 79
SPY_STATUS2_ETHERNET_FCS_AVAILABLE (*in module ics*), 79
SPY_STATUS2_ETHERNET_FRAME_TOO_SHORT (*in module ics*), 79
SPY_STATUS2_ETHERNET_NO_PADDING (*in module ics*), 79
SPY_STATUS2_ETHERNET_PREEMPTION_ENABLED (*in module ics*), 79
SPY_STATUS2_FLEXRAY_NO_CRC (*in module ics*), 79
SPY_STATUS2_FLEXRAY_NO_HEADERCRC (*in module ics*), 79
SPY_STATUS2_FLEXRAY_TX_AB (*in module ics*), 79
SPY_STATUS2_FLEXRAY_TX_AB_NO_A (*in module ics*), 79
SPY_STATUS2_FLEXRAY_TX_AB_NO_B (*in module ics*), 79
SPY_STATUS2_FLEXRAY_TX_AB_NO_MATCH (*in module ics*), 79
SPY_STATUS2_GLOBAL_CHANGE (*in module ics*), 79
SPY_STATUS2_HAS_VALUE (*in module ics*), 79
SPY_STATUS2_HIGH_VOLTAGE (*in module ics*), 79
SPY_STATUS2_ISO_FRAME_ERROR (*in module ics*), 79
SPY_STATUS2_ISO_OVERFLOW_ERROR (*in module ics*), 79
SPY_STATUS2_ISO_PARITY_ERROR (*in module ics*), 79
SPY_STATUS2_LIN_ERR_MSG_ID_PARITY (*in module ics*), 79
SPY_STATUS2_LIN_ERR_RX_BREAK_NOT_0 (*in module ics*), 79
SPY_STATUS2_LIN_ERR_RX_BREAK_TOO_SHORT (*in module ics*), 79
SPY_STATUS2_LIN_ERR_RX_DATA_GREATER_8 (*in module ics*), 79
SPY_STATUS2_LIN_ERR_RX_SYNC_NOT_55 (*in module ics*), 79
SPY_STATUS2_LIN_ERR_TX_RX_MISMATCH (*in module ics*), 79
SPY_STATUS2_LIN_ID_FRAME_ERROR (*in module ics*), 79
SPY_STATUS2_LIN_NO_SLAVE_DATA (*in module ics*), 79
SPY_STATUS2_LIN_SLAVE_BYTE_ERROR (*in module ics*), 79
SPY_STATUS2_LIN_SYNC_FRAME_ERROR (*in module ics*), 79
SPY_STATUS2_LONG_MESSAGE (*in module ics*), 80
SPY_STATUS2_MOST_CHANGED_PAR (*in module ics*), 80
SPY_STATUS2_MOST_CONTROL_DATA (*in module ics*), 80
SPY_STATUS2_MOST_I2S_DUMP (*in module ics*), 80
SPY_STATUS2_MOST_LOW_LEVEL (*in module ics*), 80
SPY_STATUS2_MOST_MHP_CONTROL_DATA (*in module ics*), 80
SPY_STATUS2_MOST_MHP_USER_DATA (*in module ics*), 80

- SPY_STATUS2_MOST_MOST150 (in module ics), 80
 SPY_STATUS2_MOST_MOST50 (in module ics), 80
 SPY_STATUS2_MOST_PACKET_DATA (in module ics), 80
 SPY_STATUS2_MOST_TOO_SHORT (in module ics), 80
 SPY_STATUS2_RX_TIMEOUT_ERROR (in module ics), 80
 SPY_STATUS2_VALUE_IS_BOOLEAN (in module ics), 80
 SPY_STATUS3_CANFD_BRS (in module ics), 80
 SPY_STATUS3_CANFD_ESI (in module ics), 80
 SPY_STATUS3_CANFD_FDF (in module ics), 80
 SPY_STATUS3_CANFD_IDE (in module ics), 80
 SPY_STATUS3_CANFD_RTR (in module ics), 80
 SPY_STATUS3_LIN_JUST_BREAK_SYNC (in module ics), 80
 SPY_STATUS3_LIN_ONLY_UPDATE_SLAVE_TABLE (in module ics), 80
 SPY_STATUS3_LIN_SLAVE_DATA_TOO_SHORT (in module ics), 80
 SPY_STATUS_ANALOG_DIGITAL_INPUT (in module ics), 80
 SPY_STATUS_AUDIO_COMMENT (in module ics), 80
 SPY_STATUS_AVSI_REC_OVERFLOW (in module ics), 80
 SPY_STATUS_BAD_MESSAGE_BIT_TIME_ERROR (in module ics), 80
 SPY_STATUS_BREAK (in module ics), 80
 SPY_STATUS_BUS_RECOVERED (in module ics), 80
 SPY_STATUS_BUS_SHORTED_GND (in module ics), 80
 SPY_STATUS_BUS_SHORTED_PLUS (in module ics), 80
 SPY_STATUS_CAN_BUS_OFF (in module ics), 80
 SPY_STATUS_CAN_ERROR_PASSIVE (in module ics), 80
 SPY_STATUS_CANFD (in module ics), 80
 SPY_STATUS_CHECKSUM_ERROR (in module ics), 80
 SPY_STATUS_COMM_IN_OVERFLOW (in module ics), 80
 SPY_STATUS_CRC_ERROR (in module ics), 80
 SPY_STATUS_EXPECTED_LEN_MISMATCH (in module ics), 80
 SPY_STATUS_EXTENDED (in module ics), 81
 SPY_STATUS_FLEXRAY_PDU (in module ics), 81
 SPY_STATUS_FLEXRAY_PDU_NO_UPDATE_BIT (in module ics), 81
 SPY_STATUS_FLEXRAY_PDU_UPDATE_BIT_SET (in module ics), 81
 SPY_STATUS_GLOBAL_ERR (in module ics), 81
 SPY_STATUS_GPS_DATA (in module ics), 81
 SPY_STATUS_HEADERCRC_ERROR (in module ics), 81
 SPY_STATUS_HIGH_SPEED (in module ics), 81
 SPY_STATUS_INCOMPLETE_FRAME (in module ics), 81
 SPY_STATUS_INIT_MESSAGE (in module ics), 81
 SPY_STATUS_LIN_MASTER (in module ics), 81
 SPY_STATUS_LOST_ARBITRATION (in module ics), 81
 SPY_STATUS_MSG_NO_MATCH (in module ics), 81
 SPY_STATUS_NETWORK_MESSAGE_TYPE (in module ics), 81
 SPY_STATUS_PDU (in module ics), 81
 SPY_STATUS_REMOTE_FRAME (in module ics), 81
 SPY_STATUS_TEST_TRIGGER (in module ics), 81
 SPY_STATUS_TEXT_COMMENT (in module ics), 81
 SPY_STATUS_TX_MSG (in module ics), 81
 SPY_STATUS_TX_NOMATCH (in module ics), 81
 SPY_STATUS_UNDEFINED_ERROR (in module ics), 81
 SPY_STATUS_VSI_IFR_CRC_BIT (in module ics), 81
 SPY_STATUS_VSI_TX_UNDERRUN (in module ics), 81
 SPY_STATUS_XTD_FRAME (in module ics), 81
 SpyMessage (class in ics), 34
 SpyMessageJ1850 (class in ics), 35
 StatusBitField (ics.SpyMessage attribute), 34
 StatusBitField (ics.SpyMessageJ1850 attribute), 36
 StatusBitField2 (ics.SpyMessage attribute), 34
 StatusBitField2 (ics.SpyMessageJ1850 attribute), 36
 StatusBitField3 (ics.SpyMessage attribute), 34
 StatusBitField3 (ics.SpyMessageJ1850 attribute), 36
 StatusBitField4 (ics.SpyMessage attribute), 35
 StatusBitField4 (ics.SpyMessageJ1850 attribute), 36
 stMin (ics.CmISO157652RxMessage attribute), 22
 stMin (ics.CmISO157652TxMessage attribute), 22
 stop_bits (ics.UartSettings attribute), 37
 swcan (ics.FireSettings attribute), 28
 swcan1 (ics.CyanSettings attribute), 25
 swcan1 (ics.RadGalaxySettings attribute), 32
 swcan1 (ics.VividCANSettings attribute), 41
 swcan2 (ics.CyanSettings attribute), 25
 swcan2 (ics.RadGalaxySettings attribute), 32
 SWCAN_AUTOSWITCH_DISABLED (in module ics), 81
 SWCAN_AUTOSWITCH_DISABLED_RESISTOR_ENABLED (in module ics), 81
 SWCAN_AUTOSWITCH_NO_RESISTOR (in module ics), 81
 SWCAN_AUTOSWITCH_WITH_RESISTOR (in module ics), 81
 SWCAN_SETTINGS_SIZE (in module ics), 81

SWCanSettings (class in ics), 33

T

tapPair0 (ics.OpEthGeneralSettings attribute), 29
tapPair1 (ics.OpEthGeneralSettings attribute), 29
tapPair2 (ics.OpEthGeneralSettings attribute), 29
tapPair3 (ics.OpEthGeneralSettings attribute), 29
tapPair4 (ics.OpEthGeneralSettings attribute), 29
tapPair5 (ics.OpEthGeneralSettings attribute), 29
termination_enables (ics.CyanSettings attribute), 25
termination_enables (ics.Vcan412Settings attribute), 38
termination_enables (ics.Vcan4Settings attribute), 39
termination_enables (ics.VividCANSettings attribute), 41
text_api (ics.CyanSettings attribute), 25
text_api (ics.FireSettings attribute), 28
text_api (ics.RadGalaxySettings attribute), 32
text_api (ics.RadStar2Settings attribute), 33
text_api (ics.Vcan412Settings attribute), 38
text_api (ics.Vcan4Settings attribute), 39
TextApiSettings (class in ics), 36
time_500us (ics.Iso9141Keyword2000InitSteps attribute), 28
TimeHardware (ics.SpyMessage attribute), 35
TimeHardware (ics.SpyMessageJ1850 attribute), 36
TimeHardware2 (ics.SpyMessage attribute), 35
TimeHardware2 (ics.SpyMessageJ1850 attribute), 36
TimeStampHardwareID (ics.SpyMessage attribute), 35
TimeStampHardwareID (ics.SpyMessageJ1850 attribute), 36
TimeStampSystemID (ics.SpyMessage attribute), 35
TimeStampSystemID (ics.SpyMessageJ1850 attribute), 36
TimesyncSettings (class in ics), 37
timeSyncSettings (ics.RadStar2Settings attribute), 33
TimeSystem (ics.SpyMessage attribute), 35
TimeSystem (ics.SpyMessageJ1850 attribute), 36
TimeSystem2 (ics.SpyMessage attribute), 35
TimeSystem2 (ics.SpyMessageJ1850 attribute), 36
TqProp (ics.CanSettings attribute), 21
TqProp (ics.SWCanSettings attribute), 33
TqSeg1 (ics.CanSettings attribute), 21
TqSeg1 (ics.SWCanSettings attribute), 34
TqSeg2 (ics.CanSettings attribute), 21
TqSeg2 (ics.SWCanSettings attribute), 34
TqSync (ics.CanSettings attribute), 21
TqSync (ics.SWCanSettings attribute), 34
transceiver_mode (ics.CanSettings attribute), 21

transceiver_mode (ics.SWCanSettings attribute), 34
transmit_messages () (in module ics), 68
tx_index (ics.CmISO157652TxMessage attribute), 22
TxMessages () (in module ics), 48

U

uart (ics.FireSettings attribute), 28
uart2 (ics.FireSettings attribute), 28
UART_SETTINGS_SIZE (in module ics), 81
UartSettings (class in ics), 37
ucConfigMode (ics.OpEthSettings attribute), 30
ucInterfaceType (ics.OpEthGeneralSettings attribute), 29
usbHostPowerEnabled (ics.Fire2DeviceStatus attribute), 26
USE_TQ (in module ics), 81

V

validate_hobject () (in module ics), 69
ValidateHObject () (in module ics), 48
vcan3 (ics.DeviceSettings attribute), 25
Vcan3Settings (class in ics), 37
vcan4 (ics.DeviceSettings attribute), 25
Vcan412Settings (class in ics), 37
vcan4_12 (ics.DeviceSettings attribute), 25
Vcan4DeviceStatus (class in ics), 38
Vcan4Settings (class in ics), 38
vcan4Status (ics.IcsDeviceStatus attribute), 28
VcanRFSettings (class in ics), 39
vividcan (ics.DeviceSettings attribute), 26
VividCANSettings (class in ics), 40
vnetBits (ics.FireSettings attribute), 28
VNETBITS_FEATURE_ANDROID_MSGS (in module ics), 81
VNETBITS_FEATURE_DISABLE_USB_CHECK (in module ics), 81
vs_netid (ics.CmISO157652RxMessage attribute), 22
vs_netid (ics.CmISO157652TxMessage attribute), 22

W

write_sdcard () (in module ics), 69
WriteSDCard () (in module ics), 48