
naqslab_devices

Release 0.2.7

dihm

Mar 04, 2020

CONTENTS:

1	Installation	3
2	Usage	5
3	Devices	7
3.1	labscript Primitive Subclasses	7
3.2	VISA	10
3.3	SignalGenerator	15
3.4	PulseBlasterESRPro300	66
3.5	PulseBlaster_No_DDS_200	71
3.6	KeysightXSeries	76
3.7	NovaTechDDS	82
3.8	SR865	95
3.9	TektronixTDS	100
3.10	KeysightDCSupply	105
4	Building Documentation	113
4.1	Sphinx Environment	113
4.2	Sphinx Build	113
4.3	Adding Documentation	113
5	Indices and tables	115
6	Credits	117
	Python Module Index	119
	Index	121

This library is a collection of third-party devices for the labsript_suite experimental control system.

INSTALLATION

Prerequisite: `labscript_devices >= 2.2.0`

Clone this repository into the `labscript_suite` directory. Typically into `userlib` or the main directory level.

Next, modify your `labconfig.ini` file to recognize the module by adding the following entry to the `[DEFAULT]` block:

```
user_devices = naqslab_devices
```

If more than one module of 3rd party devices are used, put all module names as a comma separated list.

CHAPTER TWO

USAGE

Invoke in labscript scripts just like other labscript devices:

```
from naqslab_devices import ScopeChannel
from naqslab_devices.KeysightXSeries.labscript_device import KeysightXScope
```

Details for how to use each device are contained in the *detailed documentation* listings.

DEVICES

Directory of all device classes in this repository.

The labscript primitive subclasses are derivatives of the labscript-provided children classes used by devices in this repository.

There are two parent classes that are not directly used, but rather provide templates for creating new devices. First is the *VISA* class that templates communication with devices through the VISA communication protocol. This uses the *PyVISA python wrapper*. The second is the *SignalGenerator* class that uses the *VISA* class to template CW frequency generators.

There are two thin subclasses of the labscript_devices.PulseBlaster_No_DDS class: *PulseBlasterESRPro300* and *PulseBlaster_No_DDS_200*. They exist simply to enforce the correct core clock frequency and clock limits without any other change in functionality from the parent.

Other device classes control particular series of devices and implement functional control of their hardware to varying degrees. In general, the design philosophy is that if the device class does not set an option, it will not be interfered with when using the device class to control the instrument. This means that custom settings and configurations of each device can be used by setting them manually at the device front panel without the device class interfering.

3.1 labscript Primitive Subclasses

3.1.1 Overview

These are subclasses of labscript primitive objects for specific use with the devices in this module.

StaticFreqAmp is used by the *NovaTechDDS* devices. *ScopeChannel* is used by the KeysightXSeries and TektronixTDS oscilloscope classes. *CounterScopeChannel* is used exclusively by the KeysightXSeries oscilloscopes.

<i>StaticFreqAmp</i>	This instantiates a static frequency output channel.
<i>ScopeChannel</i>	This instantiates a scope channel to acquire during a buffered shot.
<i>CounterScopeChannel</i>	This instantiates a counter scope channel to acquire during a buffered shot.

3.1.2 Detailed Documentation

class naqslab_devices.**ScopeChannel** (*name*, *parent_device*, *connection*)

Bases: labscript.labscript.AnalogIn

This instantiates a scope channel to acquire during a buffered shot.

Parameters

- **name** (*str*) – Name to assign channel

- **parent_device** (*obj*) – Handle to parent device
- **connection** (*str*) – Which physical scope channel is acquiring. Generally of the form ‘Channel n’ where n is the channel label.

description = 'Scope Acquisition Channel Class'

acquire ()

Inform BLACS to save data from this channel.

Note that the parent_device controls when the acquisition trigger is sent.

add_device (*device*)

allowed_children = None

generate_code (*hdf5_file*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val, ...} where each **val** is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

class naqslab_devices.**CounterScopeChannel** (*name, parent_device, connection*)

Bases: [naqslab_devices.ScopeChannel](#)

This instantiates a counter scope channel to acquire during a buffered shot.

Parameters

- **name** (*str*) – Name to assign channel
- **parent_device** (*obj*) – Handle to parent device
- **connection** (*str*) – Which physical scope channel is acquiring. Generally of the form ‘Channel n’ where n is the channel label.

description = 'Scope Acquisition Channel Class with Pulse Counting'

count (*typ, pol*)

Register a pulse counter operation for this channel.

Parameters

- **typ** (*str*) – count ‘pulse’ or ‘edge’
- **pol** (*str*) – reference to ‘pos’ or ‘neg’ edges

acquire ()

Inform BLACS to save data from this channel.

Note that the `parent_device` controls when the acquisition trigger is sent.

add_device (*device*)

allowed_children = None

generate_code (*hdf5_file*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into `properties_dict`

property_names is a dictionary **{key:val, ...}** where each **val** is a list [`var1`, `var2`, ...] of variables to be pulled from `properties_dict` and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

```
class naqslab_devices.StaticFreqAmp (name, parent_device, connection, freq_limits=None,
                                     freq_conv_class=None, freq_conv_params={},
                                     amp_limits=None, amp_conv_class=None,
                                     amp_conv_params={})
```

Bases: `labscript.labscript.StaticDDS`

This instantiates a static frequency output channel.

Frequency and amplitude limits set here will supercede those dictated by the device class, but only when compiling a shot with `runmanager`. Static update limits are enforced by the BLACS Tab for the parent device.

Parameters

- **name** (*str*) – Name to assign output channel
- **parent_device** (*obj*) – Handle to parent device
- **connection** (*str*) – Which physical channel to use on parent device. Typically only 'Channel 0' is available.
- **freq_limits** (*tuple*) – Set (min,max) output frequencies in BLACS front panel units.
- **freq_conv_class** (*obj*) – Custom conversion class to use
- **freq_conv_params** (*dict*) – Parameters to conversion class
- **amp_limits** (*tuple*) – Set (min,max) output amplitude in BLACS front panel units.
- **amp_conv_class** (*obj*) – Custom conversion class to use

- **amp_conv_params** (*dict*) – Parameters to conversion class

description = 'Frequency Source class for Signal Generators'

allowed_children = [`<class 'labscript.labscript.StaticAnalogQuantity'>`]

setphase (*value*, *units=None*)
Overridden from StaticDDS so as not to provide phase control, which is generally not supported by SignalGenerator devices.

enable ()
overridden from StaticDDS so as not to provide time resolution - output can be enabled or disabled only at the start of the shot

disable ()
overridden from StaticDDS so as not to provide time resolution - output can be enabled or disabled only at the start of the shot

add_device (*device*)

generate_code (*hdf5_file*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)
Get all properties in location

If location is None we return all keys

get_property (*name*, *location=None*, **args*, ***kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict*, *property_names*, *overwrite=False*)
Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name*, *value*, *location=None*, *overwrite=False*)

setamp (*value*, *units=None*)

setfreq (*value*, *units=None*)

property t0
The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

3.2 VISA

3.2.1 Overview

This parent class establishes the standard communication protocol for all devices that use the VISA communication library. It relies on the [PyVISA python wrapper](#).

<code>naqslab_devices.VISA.blacs_tab</code>	Boiler plate blacs_tab for VISA instruments.
<code>naqslab_devices.VISA.blacs_worker</code>	Boiler plate BLACS_worker for VISA instruments.

Continued on next page

Table 2 – continued from previous page

<code>naqslab_devices.VISA.labscrip_device</code>	Boiler plate labscrip_device for VISA instruments.
<code>naqslab_devices.VISA.register_classes</code>	Sets which BLACS_tab belongs to each labscrip device.

3.2.2 Detailed Documentation of naqslab_devices.VISA

VISA.blacs_tab

Boiler plate blacs_tab for VISA instruments.

Defines the common STBstatus.ui widget all devices use to report their current status.

class naqslab_devices.VISA.blacs_tab.VISATab (*args, **kwargs)

Bases: blacs.device_base_class.DeviceTab

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

status_byte_labels = {'bit 0': 'bit 0 label', 'bit 1': 'bit 1 label', 'bit 2': 'bit 2 label', ...}

status_widget = 'STBstatus.ui'

STBui_path = 'C:\\\\labscrip_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in __init__

status_monitor (*args, **kwargs)

send_clear (*args, **kwargs)

ICON_BUSY = ':/qtutils/fugue/hourglass'

ICON_ERROR = ':/qtutils/fugue/exclamation'

ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'

ICON_OK = ':/qtutils/fugue/tick'

abort_buffered (*args, **kwargs)

abort_transition_to_buffered (*args, **kwargs)

add_secondary_worker (worker)

auto_create_widgets ()

auto_place_widgets (*args)

check_remote_values (*args, **kwargs)

check_time ()

clean_ui_on_restart ()

close_tab (finalise=True)

Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations

connect_restart_receiver (function)

continue_restart (currentpage)

Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (analog_properties)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name*, *WorkerClass*, *workerargs=None*)

Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_built_in_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (**args*, ***kwargs*)

program_device_properties (**args*, ***kwargs*)


```

queue_work (worker_process, worker_function, *args, **kwargs)

restart (*args)

restore_built_in_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (data)

set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively

set_terminal_visible (visible)

shutdown_workers (*args, **kwargs)

start_run (*args, **kwargs)

property state

statemachine_timeout_add (delay, statefunction, *args, **kwargs)

statemachine_timeout_remove (statefunction)

statemachine_timeout_remove_all ()

supports_remote_value_check (support)

supports_smart_programming (support)

transition_to_buffered (*args, **kwargs)

transition_to_manual (*args, **kwargs)

update_from_settings (settings)

```

VISA.blacs_worker

Boiler plate BLACS_worker for VISA instruments.

Inheritors use the same communication protocol, but override the command syntax.

```

class naqslab_devices.VISA.blacs_worker.VISAWorker (*args, **kwargs)
    Bases: blacs.tab_base_classes.Worker

    init ()
        Initializes basic worker and opens VISA connection to device.

        Default connection timeout is 2 seconds

    check_remote_values ()

    convert_register (register)
        Converts returned register value to dict of bools

        Parameters register (int) – Status register value returned from read_stb

        Returns Status byte dictionary as formatted in VISATab

        Return type dict

    check_status ()
        Reads the Status Byte Register of the VISA device.

        Returns Status byte dictionary as formatted in VISATab

        Return type dict

    program_manual (front_panel_values)
        Over-ride this method if remote programming is supported.

```

Returns `VISAWorker.check_remote_values()`

clear (*value*)

Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in transition_to_manual method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

abort_transition_to_buffered ()

Special abort shot configuration code belongs here.

abort_buffered ()

Special abort shot code belongs here.

transition_to_manual (*abort=False*)

Simple transition_to_manual method where no data is saved.

shutdown ()

Closes VISA connection to device.

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was interrupted before the child was started, otherwise self.child will be the child Popen object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call Popen.terminate() and Popen.wait() on it.

VISA.labscript_device

Boiler plate labscript_device for VISA instruments.

class naqslab_devices.VISA.labscript_device.VISA (*name, parent_device, VISA_name, **kwargs*)

Bases: labscript.labscript.Device

Base VISA labscript_device class.

Inheritors should call VISA.__init__() in their own __init__() method. Generate_code must be overridden.

Parameters

- **name** (*str*) – name of device in connectiontable

- **parent_device** (*obj*) – Handle to any parent device.
- **VISA_name** (*str*) – Can be full VISA connection string or NI-MAX alias.

description = 'VISA Compatible Instrument'

allowed_children = []

generate_code (*hdf5_file*)

Method to generate instructions for blacs_worker to program device.

Must be over-ridden.

add_device (*device*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val, ...} where each **val** is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

VISA.register_classes

Sets which BLACS_tab belongs to each labscript device.

3.3 SignalGenerator

3.3.1 Overview

This is a parent class for static RF Signal Generators with frequency and amplitude control and use the IEEE 488 or SCPI command protocols. Device communication is handled using the VISA communication protocol.

naqslab_devices.SignalGenerator.

Models

naqslab_devices.SignalGenerator.

blacs_tab

naqslab_devices.SignalGenerator.

blacs_worker

Continued on next page

Table 3 – continued from previous page

<code>naqslab_devices.SignalGenerator.labscript_device</code>	
<code>naqslab_devices.SignalGenerator.register_classes</code>	Configures which BLACS_tab goes to which labscript_device.

3.3.2 Models

Currently covered models include:

<code>HP_8642A</code>
<code>HP_8643A</code>
<code>HP_8648</code>
<code>RS_SMA100B</code>
<code>RS_SMF100A</code>
<code>RS_SMHU</code>
<code>KeysightSigGens</code>

3.3.3 Adding a Signal Generator

In order to add a new model of signal generator one must:

1. Add a subclass `SignalGenerator.labscript_device` in `SignalGenerator.Models` which provides the operational limits.
2. Subclass `SignalGenerator.blacs_tab` and `SignalGenerator.blacs_worker` with the operational details for communicating with the device (namely command syntax and status byte definitions).
3. Add an entry in `SignalGenerator.register_classes` that links the `labscript_device` to the correct `blacs_tab`. Note that multiple `labscript_devices` can use the same `blacs_tab/blacs_worker`.

3.3.4 Detailed Documentation

```
class naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab(*args,
                                                                **kwargs)
```

Bases: `naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

```
base_units = {'amp': 'dBm', 'freq': 'MHz'}
```

```
base_min = {'amp': -140, 'freq': 0.1}
```

```
base_max = {'amp': 20, 'freq': 1057.5}
```

```
base_step = {'amp': 0.1, 'freq': 1}
```

```
base_decimals = {'amp': 1, 'freq': 6}
```

```
status_byte_labels = {'bit 0': 'End of Sweep', 'bit 1': 'Hardware Error', 'bit 2': 'Out of Range'}
```

```
ICON_BUSY = ':/qtutils/fugue/hourglass'
```

```
ICON_ERROR = ':/qtutils/fugue/exclamation'
```

```
ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
```

```
ICON_OK = ':/qtutils/fugue/tick'
```

```
STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
```

abort_buffered (**args, **kwargs*)
abort_transition_to_buffered (**args, **kwargs*)
add_secondary_worker (*worker*)
auto_create_widgets ()
auto_place_widgets (**args*)
check_remote_values (**args, **kwargs*)
check_time ()
clean_ui_on_restart ()
close_tab (*finalise=True*)
 Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call *finalise_close_tab()* to perform these potentially blocking operations
connect_restart_receiver (*function*)
continue_restart (*currentpage*)
 Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs (*analog_properties*)
create_analog_widgets (*channel_properties*)
create_dds_outputs (*dds_properties*)
create_dds_widgets (*channel_properties*)
create_device_properties (*device_properties*)
create_digital_outputs (*digital_properties*)
create_digital_widgets (*channel_properties*)
create_image_outputs (*image_properties*)
create_image_widgets (*channel_properties*)
create_property_widgets (*device_properties*)
create_worker (*name, WorkerClass, workerargs=None*)
 Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.
property device_name
disconnect_restart_receiver (*function*)
property error_message
finalise_close_tab (*currentpage*)
finalise_restart (*currentpage*)
property force_full_buffered_reprogram
get_all_save_data ()
get_builtin_save_data ()
 Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

`get_channel` (*channel*)

`get_child_from_connection_table` (*parent_device_name*, *port*)

`get_front_panel_properties` ()

`get_front_panel_values` ()

`get_property` (*property*)

`get_save_data` ()

`get_tab_layout` ()

`hide_error` ()

`initialise_GUI` ()
Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

`initialise_workers` ()

`mainloop` ()

`property mode`

`on_force_full_buffered_reprogram` ()

`on_resolve_value_inconsistency` ()

`property primary_worker`

`program_device` (**args*, ***kwargs*)

`program_device_properties` (**args*, ***kwargs*)

`queue_work` (*worker_process*, *worker_function*, **args*, ***kwargs*)

`restart` (**args*)

`restore_builtin_save_data` (*data*)
Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

`restore_save_data` (*data*)

`send_clear` (**args*, ***kwargs*)

`set_tab_icon_and_colour` ()
Set the tab icon and the colour of its text to the values of `self._tab_icon` and `self._tab_text_colour` respectively

`set_terminal_visible` (*visible*)

`shutdown_workers` (**args*, ***kwargs*)

`start_run` (**args*, ***kwargs*)

`property state`

`statemachine_timeout_add` (*delay*, *statefunction*, **args*, ***kwargs*)

`statemachine_timeout_remove` (*statefunction*)

`statemachine_timeout_remove_all` ()

`status_monitor` (**args*, ***kwargs*)

`status_widget` = 'STBstatus.ui'

`supports_remote_value_check` (*support*)

`supports_smart_programming` (*support*)

`transition_to_buffered` (**args*, ***kwargs*)

`transition_to_manual` (**args*, ***kwargs*)

```

    update_from_settings (settings)

class naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
    scale_factor = 1000000.0
    amp_scale_factor = 1.0
    freq_write_string = 'FR {:.0f} HZ'
    freq_query_string = 'FROA'
    freq_parser (freq_string)
        Frequency Query string parser for HP8642A freq_string format is FR sddddddddd.0 HZ Returns float
        in instrument units, Hz (i.e. needs scaling to base_units)
    amp_write_string = 'AP {:.1f} DM'
    amp_query_string = 'APOA'
    amp_parser (amp_string)
        Amplitude Query string parser for HP8642A amp_string format is AP sddd.d DM Returns float in
        instrument units, dBm
    abort_buffered ()
        Special abort shot code belongs here.
    abort_transition_to_buffered ()
        Special abort shot configuration code belongs here.
    check_remote_values ()
    check_status ()
        Reads the Status Byte Register of the VISA device.
            Returns Status byte dictionary as formatted in VISATab
            Return type dict
    clear (value)
        Sends standard *CLR to clear registers of device.
            Parameters value (bool) – value of Clear button in STBstatus.ui widget
    convert_register (register)
        Converts returned register value to dict of bools
            Parameters register (int) – Status register value returned from read_stb
            Returns Status byte dictionary as formatted in VISATab
            Return type dict
    init ()
        Initializes basic worker and opens VISA connection to device.
        Default connection timeout is 2 seconds
    interrupt_startup (reason='Process.interrupt_startup() called')
        Called from the parent process. Interrupt all blocking operations on starting the child process, causing
        Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was
        interrupted before the child was started, otherwise self.child will be the child Popen object, which
        could be at any stage of setting up its connection with the parent. This method may be called multiple
        times without raising an exception, it will simply do nothing if startup has previously been interrupted
    mainloop ()
    program_manual (front_panel_values)
        Over-ride this method if remote programming is supported.

```

Returns `VISAWorker.check_remote_values()`

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in `transition_to_manual` method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple `transition_to_manual` method where no data is saved.

class `naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` (**args, **kwargs*)

Bases: `naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

`base_units = {'amp': 'dBm', 'freq': 'MHz'}`

`base_min = {'amp': -137, 'freq': 0.26}`

`base_max = {'amp': 13, 'freq': 1030}`

`base_step = {'amp': 0.1, 'freq': 1}`

`base_decimals = {'amp': 2, 'freq': 8}`

`status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q`

`ICON_BUSY = ':/qtutils/fugue/hourglass'`

`ICON_ERROR = ':/qtutils/fugue/exclamation'`

`ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'`

`ICON_OK = ':/qtutils/fugue/tick'`

`STBui_path = 'C:\\\\labscrip_t_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'`

`abort_buffered(*args, **kwargs)`

`abort_transition_to_buffered(*args, **kwargs)`

`add_secondary_worker(worker)`

`auto_create_widgets()`

`auto_place_widgets(*args)`

`check_remote_values(*args, **kwargs)`

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)
Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call *finalise_close_tab()* to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)
Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)
Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()
Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

```
get_tab_layout ()
hide_error ()
initialise_GUI ()
    Loads the standard STBstatus.ui widget and sets the worker defined in __init__
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_builtin_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
send_clear (*args, **kwargs)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
status_monitor (*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)

class naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
    scale_factor = 1000000.0
    amp_scale_factor = 1.0
```

init()
Calls parent init and sends device specific initialization commands

freq_write_string = 'FREQ:CW {:.2f} HZ'

freq_query_string = 'FREQ:CW?'

freq_parser (*freq_string*)
Frequency Query string parser for HP8643A *freq_string* format is float, in Hz Returns float in instrument units, Hz (i.e. needs scaling to *base_units*)

amp_write_string = 'AMPL:LEV {:.2f} DBM'

amp_query_string = 'AMPL:LEV?'

amp_parser (*amp_string*)
Amplitude Query string parser for HP8643A *amp_string* format is float in configured units (dBm by default) Returns float in instrument units, dBm

check_status()
Reads the Status Byte Register of the VISA device.
Returns Status byte dictionary as formatted in VISATab
Return type dict

abort_buffered()
Special abort shot code belongs here.

abort_transition_to_buffered()
Special abort shot configuration code belongs here.

check_remote_values()

clear (*value*)
Sends standard *CLR to clear registers of device.
Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register (*register*)
Converts returned register value to dict of bools
Parameters **register** (*int*) – Status register value returned from `read_stb`
Returns Status byte dictionary as formatted in VISATab
Return type dict

interrupt_startup (*reason*=*'Process.interrupt_startup() called'*)
Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop()

program_manual (*front_panel_values*)
Over-ride this method if remote programming is supported.
Returns `VISAWorker.check_remote_values()`

run (*worker_name*, *device_name*, *extraargs*)
The method that gets called in the subprocess. To be overridden by subclasses

shutdown()
Closes VISA connection to device.

start (**args*, ***kwargs*)
Call in the parent process to start a subprocess. Passes *args* and *kwargs* to the `run()` method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call Popen.terminate() and Popen.wait() on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in transition_to_manual method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple transition_to_manual method where no data is saved.

class naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab (**args, **kwargs*)

Bases: *naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab*

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

base_units = {'amp': 'dBm', 'freq': 'MHz'}

base_min = {'amp': -136, 'freq': 0.1}

base_max = {'amp': 20, 'freq': 1000}

base_step = {'amp': 1, 'freq': 1}

base_decimals = {'amp': 1, 'freq': 9}

status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q

ICON_BUSY = ':/qtutils/fugue/hourglass'

ICON_ERROR = ':/qtutils/fugue/exclamation'

ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'

ICON_OK = ':/qtutils/fugue/tick'

STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'

abort_buffered (**args, **kwargs*)

abort_transition_to_buffered (**args, **kwargs*)

add_secondary_worker (*worker*)

auto_create_widgets ()

auto_place_widgets (**args*)

check_remote_values (**args, **kwargs*)

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)

Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)

Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.

Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name*, *WorkerClass*, *workerargs=None*)

Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

initialise_workers ()

mainloop ()

property mode

```
on_force_full_buffered_reprogram()
on_resolve_value_inconsistency()
property primary_worker
program_device(*args, **kwargs)
program_device_properties(*args, **kwargs)
queue_work(worker_process, worker_function, *args, **kwargs)
restart(*args)
restore_builtin_save_data(data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data(data)
send_clear(*args, **kwargs)
set_tab_icon_and_colour()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible(visible)
shutdown_workers(*args, **kwargs)
start_run(*args, **kwargs)
property state
statemachine_timeout_add(delay, statefunction, *args, **kwargs)
statemachine_timeout_remove(statefunction)
statemachine_timeout_remove_all()
status_monitor(*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check(support)
supports_smart_programming(support)
transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)

class naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab(*args,
                                                             **kwargs)
    Bases: naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab

    You MUST override this method in order to define the device worker. You then call this parent method to
    finish initialization.

    base_min = {'amp': -136, 'freq': 0.009}
    base_max = {'amp': 20, 'freq': 2000}
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
    abort_buffered(*args, **kwargs)
```

```

abort_transition_to_buffered (*args, **kwargs)
add_secondary_worker (worker)
auto_create_widgets ()
auto_place_widgets (*args)
base_decimals = {'amp': 1, 'freq': 9}
base_step = {'amp': 1, 'freq': 1}
base_units = {'amp': 'dBm', 'freq': 'MHz'}
check_remote_values (*args, **kwargs)
check_time ()
clean_ui_on_restart ()
close_tab (finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver (function)
continue_restart (currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs (analog_properties)
create_analog_widgets (channel_properties)
create_dds_outputs (dds_properties)
create_dds_widgets (channel_properties)
create_device_properties (device_properties)
create_digital_outputs (digital_properties)
create_digital_widgets (channel_properties)
create_image_outputs (image_properties)
create_image_widgets (channel_properties)
create_property_widgets (device_properties)
create_worker (name, WorkerClass, workerargs=None)
    Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.
property device_name
disconnect_restart_receiver (function)
property error_message
finalise_close_tab (currentpage)
finalise_restart (currentpage)
property force_full_buffered_reprogram
get_all_save_data ()

```

get_builtin_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (args, **kwargs*)**

program_device_properties (args, **kwargs*)**

queue_work (*worker_process, worker_function, *args, **kwargs*)

restart (args*)**

restore_builtin_save_data (*data*)

Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (*data*)

send_clear (args, **kwargs*)**

set_tab_icon_and_colour ()

Set the tab icon and the colour of its text to the values of `self._tab_icon` and `self._tab_text_colour` respectively

set_terminal_visible (*visible*)

shutdown_workers (args, **kwargs*)**

start_run (args, **kwargs*)**

property state

statemachine_timeout_add (*delay, statefunction, *args, **kwargs*)

statemachine_timeout_remove (*statefunction*)

statemachine_timeout_remove_all ()

status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q

status_monitor (args, **kwargs*)**

status_widget = 'STBstatus.ui'

supports_remote_value_check (*support*)

supports_smart_programming (*support*)


```

transition_to_buffered(*args, **kwargs)

transition_to_manual(*args, **kwargs)

update_from_settings(settings)

class naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab(*args,
                                                             **kwargs)
    Bases: naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab

    You MUST override this method in order to define the device worker. You then call this parent method to
    finish initialization.

    base_min = {'amp': -136, 'freq': 0.009}
    base_max = {'amp': 20, 'freq': 3200}
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
    abort_buffered(*args, **kwargs)
    abort_transition_to_buffered(*args, **kwargs)
    add_secondary_worker(worker)
    auto_create_widgets()
    auto_place_widgets(*args)
    base_decimals = {'amp': 1, 'freq': 9}
    base_step = {'amp': 1, 'freq': 1}
    base_units = {'amp': 'dBm', 'freq': 'MHz'}
    check_remote_values(*args, **kwargs)
    check_time()
    clean_ui_on_restart()
    close_tab(finalise=True)
        Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not ter-
        minate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab()
        to perform these potentially blocking operations
    connect_restart_receiver(function)
    continue_restart(currentpage)
        Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.
        Calls subsequent GUI operations in the main thread once finished blocking.
    create_analog_outputs(analog_properties)
    create_analog_widgets(channel_properties)
    create_dds_outputs(dds_properties)
    create_dds_widgets(channel_properties)
    create_device_properties(device_properties)
    create_digital_outputs(digital_properties)
    create_digital_widgets(channel_properties)
    create_image_outputs(image_properties)

```

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)

Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in __init__

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (**args, **kwargs*)

program_device_properties (**args, **kwargs*)

queue_work (*worker_process, worker_function, *args, **kwargs*)

restart (**args*)

restore_builtin_save_data (*data*)

Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (*data*)

```

send_clear(*args, **kwargs)

set_tab_icon_and_colour()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively

set_terminal_visible(visible)

shutdown_workers(*args, **kwargs)

start_run(*args, **kwargs)

property state

statemachine_timeout_add(delay, statefunction, *args, **kwargs)

statemachine_timeout_remove(statefunction)

statemachine_timeout_remove_all()

status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q

status_monitor(*args, **kwargs)

status_widget = 'STBstatus.ui'

supports_remote_value_check(support)

supports_smart_programming(support)

transition_to_buffered(*args, **kwargs)

transition_to_manual(*args, **kwargs)

update_from_settings(settings)

class naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab(*args,
                                                             **kwargs)
    Bases: naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab

    You MUST override this method in order to define the device worker. You then call this parent method to
    finish initialization.

    base_min = {'amp': -136, 'freq': 0.009}

    base_max = {'amp': 20, 'freq': 4000}

    ICON_BUSY = ':/qtutils/fugue/hourglass'

    ICON_ERROR = ':/qtutils/fugue/exclamation'

    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'

    ICON_OK = ':/qtutils/fugue/tick'

    STBui_path = 'C:\\\\labscrip_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'

    abort_buffered(*args, **kwargs)

    abort_transition_to_buffered(*args, **kwargs)

    add_secondary_worker(worker)

    auto_create_widgets()

    auto_place_widgets(*args)

    base_decimals = {'amp': 1, 'freq': 9}

    base_step = {'amp': 1, 'freq': 1}

    base_units = {'amp': 'dBm', 'freq': 'MHz'}

    check_remote_values(*args, **kwargs)

    check_time()

```

clean_ui_on_restart ()

close_tab (*finalise=True*)

Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call `finalise_close_tab()` to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)

Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)

Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

```

hide_error()
initialise_GUI()
    Loads the standard STBstatus.ui widget and sets the worker defined in __init__
initialise_workers()
mainloop()
property mode
on_force_full_buffered_reprogram()
on_resolve_value_inconsistency()
property primary_worker
program_device(*args, **kwargs)
program_device_properties(*args, **kwargs)
queue_work(worker_process, worker_function, *args, **kwargs)
restart(*args)
restore_builtin_save_data(data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data(data)
send_clear(*args, **kwargs)
set_tab_icon_and_colour()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible(visible)
shutdown_workers(*args, **kwargs)
start_run(*args, **kwargs)
property state
statemachine_timeout_add(delay, statefunction, *args, **kwargs)
statemachine_timeout_remove(statefunction)
statemachine_timeout_remove_all()
status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q
status_monitor(*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check(support)
supports_smart_programming(support)
transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)

class naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker(*args,
                                                                **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
    scale_factor = 1000000.0
    amp_scale_factor = 1.0

```

init()
Calls parent init and sends device specific initialization commands

freq_write_string = 'FREQ: CW {:.3f} HZ'

freq_query_string = 'FREQ: CW?'

freq_parser (*freq_string*)
Frequency Query string parser for HP8648 freq_string format is float, in Hz Returns float in instrument units, Hz (i.e. needs scaling to base_units)

amp_write_string = 'POW: AMPL {:.1f} DBM'

amp_query_string = 'POW: AMPL?'

amp_parser (*amp_string*)
Amplitude Query string parser for HP8648 amp_string format is float in configured units (dBm by default) Returns float in instrument units, dBm

check_status()
Reads the Status Byte Register of the VISA device.

Returns Status byte dictionary as formatted in VISATab

Return type dict

abort_buffered()
Special abort shot code belongs here.

abort_transition_to_buffered()
Special abort shot configuration code belongs here.

check_remote_values()

clear (*value*)
Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register (*register*)
Converts returned register value to dict of bools

Parameters **register** (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

interrupt_startup (*reason*=*'Process.interrupt_startup() called'*)
Called from the parent process. Interrupt all blocking operations on starting the child process, causing Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was interrupted before the child was started, otherwise self.child will be the child Popen object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop()

program_manual (*front_panel_values*)
Over-ride this method if remote programming is supported.

Returns `VISAWorker.check_remote_values()`

run (*worker_name*, *device_name*, *extraargs*)
The method that gets called in the subprocess. To be overridden by subclasses

shutdown()
Closes VISA connection to device.

start (**args*, ***kwargs*)
Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call Popen.terminate() and Popen.wait() on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in transition_to_manual method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple transition_to_manual method where no data is saved.

class naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab (**args, **kwargs*)

Bases: *naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab*

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

base_units = {'amp': 'dBm', 'freq': 'GHz'}

base_min = {'amp': -145, 'freq': 8e-06}

base_max = {'amp': 35, 'freq': 20}

base_step = {'amp': 1, 'freq': 0.1}

base_decimals = {'amp': 2, 'freq': 9}

status_byte_labels = {'bit 0': 'Unused', 'bit 1': 'Unused', 'bit 2': 'Error Queue'}

ICON_BUSY = ':/qtutils/fugue/hourglass'

ICON_ERROR = ':/qtutils/fugue/exclamation'

ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'

ICON_OK = ':/qtutils/fugue/tick'

STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'

abort_buffered (**args, **kwargs*)

abort_transition_to_buffered (**args, **kwargs*)

add_secondary_worker (*worker*)

auto_create_widgets ()

auto_place_widgets (**args*)

check_remote_values (**args, **kwargs*)

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)

Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)

Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.

Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name*, *WorkerClass*, *workerargs=None*)

Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

initialise_workers ()

mainloop ()

property mode


```

on_force_full_buffered_reprogram()
on_resolve_value_inconsistency()
property primary_worker
program_device(*args, **kwargs)
program_device_properties(*args, **kwargs)
queue_work(worker_process, worker_function, *args, **kwargs)
restart(*args)
restore_builtin_save_data(data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data(data)
send_clear(*args, **kwargs)
set_tab_icon_and_colour()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible(visible)
shutdown_workers(*args, **kwargs)
start_run(*args, **kwargs)
property state
statemachine_timeout_add(delay, statefunction, *args, **kwargs)
statemachine_timeout_remove(statefunction)
statemachine_timeout_remove_all()
status_monitor(*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check(support)
supports_smart_programming(support)
transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)
class naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker(*args,
                                                                    **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
    scale_factor = 1000000000.0
    amp_scale_factor = 1.0
    freq_write_string = 'FREQ:CW {:.0f}HZ'
    freq_query_string = 'FREQ:CW?'
    freq_parser(freq_string)
        Frequency Query string parser for SMF100A freq_string format is dddddddddd Returns float in in-
        strument units, Hz (i.e. needs scaling to base_units)
    amp_write_string = 'POW:POW {:.2f}dBm'
    amp_query_string = 'POW?'
    amp_parser(amp_string)
        Amplitude Query string parser for SMF100A Returns float in instrument units, dBm

```

check_status()

Reads the Status Byte Register of the VISA device.

Returns Status byte dictionary as formatted in VISATab

Return type dict

abort_buffered()

Special abort shot code belongs here.

abort_transition_to_buffered()

Special abort shot configuration code belongs here.

check_remote_values()**clear(value)**

Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register(register)

Converts returned register value to dict of bools

Parameters **register** (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

init()

Initializes basic worker and opens VISA connection to device.

Default connection timeout is 2 seconds

interrupt_startup(reason='Process.interrupt_startup() called')

Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop()**program_manual(front_panel_values)**

Over-ride this method if remote programming is supported.

Returns `VISAWorker.check_remote_values()`

run(worker_name, device_name, extraargs)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown()

Closes VISA connection to device.

start(*args, **kwargs)

Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate(wait_timeout=None, **kwargs)

Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_buffered(device_name, h5file, initial_values, fresh)

Stores various device handles for use in `transition_to_manual` method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable

- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple transition_to_manual method where no data is saved.

class naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.**RS_SMF100ATab** (**args*,
***kwargs*)

Bases: *naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab*

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

base_units = {'amp': 'dBm', 'freq': 'GHz'}

base_min = {'amp': -26, 'freq': 1e-06}

base_max = {'amp': 18, 'freq': 22}

base_step = {'amp': 1, 'freq': 0.1}

base_decimals = {'amp': 1, 'freq': 6}

status_byte_labels = {'bit 0': 'Unused', 'bit 1': 'Unused', 'bit 2': 'Error Queue'}

ICON_BUSY = ':/qtutils/fugue/hourglass'

ICON_ERROR = ':/qtutils/fugue/exclamation'

ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'

ICON_OK = ':/qtutils/fugue/tick'

STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'

abort_buffered (**args*, ***kwargs*)

abort_transition_to_buffered (**args*, ***kwargs*)

add_secondary_worker (*worker*)

auto_create_widgets ()

auto_place_widgets (**args*)

check_remote_values (**args*, ***kwargs*)

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)

Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)

Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.

Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name*, *WorkerClass*, *workerargs=None*)
Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()
Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()
Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (**args*, ***kwargs*)

program_device_properties (**args*, ***kwargs*)

queue_work (*worker_process*, *worker_function*, **args*, ***kwargs*)

restart (**args*)

```

restore_builtin_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (data)

send_clear (*args, **kwargs)

set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively

set_terminal_visible (visible)

shutdown_workers (*args, **kwargs)

start_run (*args, **kwargs)

property state

statemachine_timeout_add (delay, statefunction, *args, **kwargs)

statemachine_timeout_remove (statefunction)

statemachine_timeout_remove_all ()

status_monitor (*args, **kwargs)

status_widget = 'STBstatus.ui'

supports_remote_value_check (support)

supports_smart_programming (support)

transition_to_buffered (*args, **kwargs)

transition_to_manual (*args, **kwargs)

update_from_settings (settings)

class naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker

    scale_factor = 1000000000.0

    amp_scale_factor = 1.0

    freq_write_string = 'FREQ:CW {:.0f}HZ'

    freq_query_string = 'FREQ?'

    freq_parser (freq_string)
        Frequency Query string parser for SMF100A freq_string format is ddddddddddd Returns float in in-
        strument units, Hz (i.e. needs scaling to base_units)

    amp_write_string = 'POW:POW {:.2f}dBm'

    amp_query_string = 'POW?'

    amp_parser (amp_string)
        Amplitude Query string parser for SMF100A Returns float in instrument units, dBm

    check_status ()
        Reads the Status Byte Register of the VISA device.

        Returns Status byte dictionary as formatted in VISATab

        Return type dict

    abort_buffered ()
        Special abort shot code belongs here.

    abort_transition_to_buffered ()
        Special abort shot configuration code belongs here.

```

check_remote_values ()

clear (*value*)

Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register (*register*)

Converts returned register value to dict of bools

Parameters **register** (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

init ()

Initializes basic worker and opens VISA connection to device.

Default connection timeout is 2 seconds

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

program_manual (*front_panel_values*)

Over-ride this method if remote programming is supported.

Returns `VISAWorker.check_remote_values()`

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in `transition_to_manual` method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple `transition_to_manual` method where no data is saved.

class `naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` (**args, **kwargs*)
Bases: `naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

```

base_units = {'amp': 'dBm', 'freq': 'MHz'}
base_min = {'amp': -140, 'freq': 0.1}
base_max = {'amp': 13, 'freq': 4320}
base_step = {'amp': 0.1, 'freq': 1}
base_decimals = {'amp': 1, 'freq': 7}
status_byte_labels = {'bit 0': 'OPC', 'bit 1': 'SRQ', 'bit 2': 'Query Error', 'bit 3': 'Error'}
ICON_BUSY = ':/qtutils/fugue/hourglass'
ICON_ERROR = ':/qtutils/fugue/exclamation'
ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
ICON_OK = ':/qtutils/fugue/tick'
STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
abort_buffered(*args, **kwargs)
abort_transition_to_buffered(*args, **kwargs)
add_secondary_worker(worker)
auto_create_widgets()
auto_place_widgets(*args)
check_remote_values(*args, **kwargs)
check_time()
clean_ui_on_restart()
close_tab(finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver(function)
continue_restart(currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.
    Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs(analog_properties)
create_analog_widgets(channel_properties)
create_dds_outputs(dds_properties)
create_dds_widgets(channel_properties)
create_device_properties(device_properties)
create_digital_outputs(digital_properties)
create_digital_widgets(channel_properties)
create_image_outputs(image_properties)
create_image_widgets(channel_properties)
create_property_widgets(device_properties)

```

create_worker (*name*, *WorkerClass*, *workerargs=None*)

Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (**args*, ***kwargs*)

program_device_properties (**args*, ***kwargs*)

queue_work (*worker_process*, *worker_function*, **args*, ***kwargs*)

restart (**args*)

restore_builtin_save_data (*data*)

Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (*data*)

send_clear (**args*, ***kwargs*)


```

set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively

set_terminal_visible (visible)

shutdown_workers (*args, **kwargs)

start_run (*args, **kwargs)

property state

statemachine_timeout_add (delay, statefunction, *args, **kwargs)

statemachine_timeout_remove (statefunction)

statemachine_timeout_remove_all ()

status_monitor (*args, **kwargs)

status_widget = 'STBstatus.ui'

supports_remote_value_check (support)

supports_smart_programming (support)

transition_to_buffered (*args, **kwargs)

transition_to_manual (*args, **kwargs)

update_from_settings (settings)

class naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker

    scale_factor = 1000000.0

    amp_scale_factor = 1.0

    init ()
        Calls parent init and sends device specific initialization commands

    freq_write_string = 'RF {:.1f}HZ'

    freq_query_string = 'RF?'

    freq_parser (freq_string)
        Frequency Query string parser for RS SMHU freq_string format is sdddddddddd.d Returns float in
        instrument units, Hz (i.e. needs scaling to base_units)

    amp_write_string = 'LEVEL:RF {:.1f}DBM'

    amp_query_string = 'LEVEL:RF?'

    amp_parser (amp_string)
        Amplitude Query string parser for RS SMHU amp_string format is sddd.d Returns float in instrument
        units, dBm

    check_status ()
        Reads the Status Byte Register of the VISA device.

        Returns Status byte dictionary as formatted in VISATab

        Return type dict

    abort_buffered ()
        Special abort shot code belongs here.

    abort_transition_to_buffered ()
        Special abort shot configuration code belongs here.

    check_remote_values ()

```

clear (*value*)

Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register (*register*)

Converts returned register value to dict of bools

Parameters **register** (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

program_manual (*front_panel_values*)

Over-ride this method if remote programming is supported.

Returns `VISAWorker.check_remote_values()`

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in `transition_to_manual` method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple `transition_to_manual` method where no data is saved.

class `naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab` (**args, **kwargs*)

Bases: `naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q

ICON_BUSY = ':/qtutils/fugue/hourglass'

```

ICON_ERROR = ':/qtutils/fugue/exclamation'
ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
ICON_OK = ':/qtutils/fugue/tick'
STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
abort_buffered (*args, **kwargs)
abort_transition_to_buffered (*args, **kwargs)
add_secondary_worker (worker)
auto_create_widgets ()
auto_place_widgets (*args)
base_decimals = {'amp': 1, 'freq': 6}
base_max = {'amp': 20, 'freq': 1057.5}
base_min = {'amp': -140, 'freq': 0.1}
base_step = {'amp': 0.1, 'freq': 1}
base_units = {'amp': 'dBm', 'freq': 'MHz'}
check_remote_values (*args, **kwargs)
check_time ()
clean_ui_on_restart ()
close_tab (finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver (function)
continue_restart (currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs (analog_properties)
create_analog_widgets (channel_properties)
create_dds_outputs (dds_properties)
create_dds_widgets (channel_properties)
create_device_properties (device_properties)
create_digital_outputs (digital_properties)
create_digital_widgets (channel_properties)
create_image_outputs (image_properties)
create_image_widgets (channel_properties)
create_property_widgets (device_properties)
create_worker (name, WorkerClass, workerargs=None)
    Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

```

property device_name
disconnect_restart_receiver (*function*)
property error_message
finalise_close_tab (*currentpage*)
finalise_restart (*currentpage*)
property force_full_buffered_reprogram
get_all_save_data ()
get_built_in_save_data ()
 Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.
get_channel (*channel*)
get_child_from_connection_table (*parent_device_name, port*)
get_front_panel_properties ()
get_front_panel_values ()
get_property (*property*)
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_GUI ()
 Loads the standard STBstatus.ui widget and sets the worker defined in __init__
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (**args, **kwargs*)
program_device_properties (**args, **kwargs*)
queue_work (*worker_process, worker_function, *args, **kwargs*)
restart (**args*)
restore_built_in_save_data (*data*)
 Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (*data*)
send_clear (**args, **kwargs*)
set_tab_icon_and_colour ()
 Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour respectively
set_terminal_visible (*visible*)
shutdown_workers (**args, **kwargs*)
start_run (**args, **kwargs*)
property state
statemachine_timeout_add (*delay, statefunction, *args, **kwargs*)

```

statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
status_monitor (*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)

class naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.
           KeysightSigGenTab

    You MUST override this method in order to define the device worker. You then call this parent method to
    finish initialization.

    base_units = {'amp': 'dBm', 'freq': 'GHz'}
    base_min = {'amp': -105, 'freq': 0.01}
    base_max = {'amp': 20, 'freq': 40.0}
    base_step = {'amp': 1, 'freq': 1}
    base_decimals = {'amp': 1, 'freq': 12}
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
    abort_buffered (*args, **kwargs)
    abort_transition_to_buffered (*args, **kwargs)
    add_secondary_worker (worker)
    auto_create_widgets ()
    auto_place_widgets (*args)
    check_remote_values (*args, **kwargs)
    check_time ()
    clean_ui_on_restart ()
    close_tab (finalise=True)
        Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate
        subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab()
        to perform these potentially blocking operations
    connect_restart_receiver (function)
    continue_restart (currentpage)
        Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.
        Calls subsequent GUI operations in the main thread once finished blocking.
    create_analog_outputs (analog_properties)

```

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name*, *WorkerClass*, *workerargs=None*)

Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_built_in_save_data ()

Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in `__init__`

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

```

program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_builton_save_data (data)
    Restore builton settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
send_clear (*args, **kwargs)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)

property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'RQC', 'bit 2': 'Q
status_monitor (*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)

class naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker (*args,
                                                                                     **kwargs)
    Bases: naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
    Generic BLACS worker for Modern Keysight/Agilent/HP CW Signal Generators

    This class defines the common frequency/amplitude read/write commands as well as general device con-
    figuration and communication. It uses the ESR byte instead of the standard STB byte for intstrument
    communications.

    The scale_factor and amp_scale_factor are taken from the labscrip device specification.

    init ()
        Calls parent init and sends device specific initialization commands

    freq_write_string = 'FREQ:CW {:.3f} HZ'
    freq_query_string = 'FREQ:CW?'
    freq_parser (freq_string)
        Frequency Query string parser for Keysight Sig Gens freq_string format is float, in Hz Returns float
        in instrument units, Hz (i.e. needs scaling to base_units)

    amp_write_string = 'POW:AMPL {:.1f} DBM'

```

amp_query_string = 'POW:AMPL?'

amp_parser (*amp_string*)

Amplitude Query string parser for Keysight Sig Gens amp_string format is float in configured units (dBm by default) Returns float in instrument units, dBm

check_status ()

Reads the Status Byte Register of the VISA device.

Returns Status byte dictionary as formatted in VISATab

Return type dict

abort_buffered ()

Special abort shot code belongs here.

abort_transition_to_buffered ()

Special abort shot configuration code belongs here.

amp_scale_factor = 1.0

check_remote_values ()

clear (*value*)

Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register (*register*)

Converts returned register value to dict of bools

Parameters **register** (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

program_manual (*front_panel_values*)

Over-ride this method if remote programming is supported.

Returns `VISAWorker.check_remote_values()`

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

scale_factor = 1000000.0

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in `transition_to_manual` method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

transition_to_manual (*abort=False*)

Simple transition_to_manual method where no data is saved.

class naqslab_devices.SignalGenerator.Models.**RS_SMF100A** (*name, VISA_name*)

Bases: *naqslab_devices.SignalGenerator.labscript_device.SignalGenerator*

VISA_name can be full VISA connection string or NI-MAX alias

description = 'Rhode & Schwarz SMF100A Signal Generator'

scale_factor = 1000000000.0

freq_limits = (100000.0, 22000000000.0)

amp_scale_factor = 1.0

amp_limits = (-26, 18)

add_device (*device*)

allowed_children = [*<class 'naqslab_devices.StaticFreqAmp'>*]

generate_code (*hdf5_file*)

Method to generate instructions for blacs_worker to program device.

Must be over-ridden.

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_amp (*data, device*)

Quantize the amplitude in units of dBm and check it's within bounds

quantise_freq (*data, device*)

Quantize the frequency in units of Hz and check it's within bounds

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val,...} where each **val** is a list [var1, var2,...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

```
class naqslab_devices.SignalGenerator.Models.RS_SMA100B(name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'Rhode & Schwarz SMA100B Signal Generator'
    scale_factor = 1000000000.0
    freq_limits = (8000.0, 20000000000.0)
    amp_scale_factor = 1.0
    amp_limits = (-145, 35)
    add_device(device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code(hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.
    get_all_children()
    get_all_outputs()
    get_properties(location=None)
        Get all properties in location
        If location is None we return all keys
    get_property(name, location=None, *args, **kwargs)
    init_device_group(hdf5_file)
    property parent_clock_line
    property pseudoclock_device
    quantise_amp(data, device)
        Quantize the amplitude in units of dBm and check it's within bounds
    quantise_freq(data, device)
        Quantize the frequency in units of Hz and check it's within bounds
    quantise_to_pseudoclock(times)
    set_properties(properties_dict, property_names, overwrite=False)
        Add one or a bunch of properties packed into properties_dict
        property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
        to be pulled from properties_dict and added to the property with name key (it's location)
    set_property(name, value, location=None, overwrite=False)
    property t0
        The earliest time output can be commanded from this device at the start of the experiment. This is
        nonzero on secondary pseudoclock devices due to triggering delays.

class naqslab_devices.SignalGenerator.Models.RS_SMHU(name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'RS SMHU Signal Generator'
    scale_factor = 1000000.0
    freq_limits = (100000.0, 4320000000.0)
    amp_scale_factor = 1.0
```

```

    amp_limits = (-140, 13)
    add_device (device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code (hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.
    get_all_children ()
    get_all_outputs ()
    get_properties (location=None)
        Get all properties in location
        If location is None we return all keys
    get_property (name, location=None, *args, **kwargs)
    init_device_group (hdf5_file)
    property parent_clock_line
    property pseudoclock_device
    quantise_amp (data, device)
        Quantize the amplitude in units of dBm and check it's within bounds
    quantise_freq (data, device)
        Quantize the frequency in units of Hz and check it's within bounds
    quantise_to_pseudoclock (times)
    set_properties (properties_dict, property_names, overwrite=False)
        Add one or a bunch of properties packed into properties_dict
        property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
        to be pulled from properties_dict and added to the property with name key (it's location)
    set_property (name, value, location=None, overwrite=False)
    property t0
        The earliest time output can be commanded from this device at the start of the experiment. This is
        nonzero on secondary pseudoclock devices due to triggering delays.
class naqslab_devices.SignalGenerator.Models.HP_8643A (name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'HP 8643A Signal Generator'
    scale_factor = 1000000.0
    freq_limits = (260000.0, 1030000000.0)
    amp_scale_factor = 1.0
    amp_limits = (-137, 13)
    add_device (device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code (hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.
    get_all_children ()

```

```
get_all_outputs ()
get_properties (location=None)
    Get all properties in location
    If location is None we return all keys
get_property (name, location=None, *args, **kwargs)
init_device_group (hdf5_file)
property parent_clock_line
property pseudoclock_device
quantise_amp (data, device)
    Quantize the amplitude in units of dBm and check it's within bounds
quantise_freq (data, device)
    Quantize the frequency in units of Hz and check it's within bounds
quantise_to_pseudoclock (times)
set_properties (properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict
    property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)
set_property (name, value, location=None, overwrite=False)
property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.
class naqslab_devices.SignalGenerator.Models.HP_8642A (name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'HP 8642A Signal Generator'
    scale_factor = 1000000.0
    freq_limits = (100000.0, 1057500000.0)
    amp_scale_factor = 1.0
    amp_limits = (-140, 20)
    add_device (device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code (hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.
    get_all_children ()
    get_all_outputs ()
    get_properties (location=None)
        Get all properties in location
        If location is None we return all keys
    get_property (name, location=None, *args, **kwargs)
    init_device_group (hdf5_file)
    property parent_clock_line
```

```

property pseudoclock_device

quantise_amp (data, device)
    Quantize the amplitude in units of dBm and check it's within bounds

quantise_freq (data, device)
    Quantize the frequency in units of Hz and check it's within bounds

quantise_to_pseudoclock (times)

set_properties (properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict

    property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)

set_property (name, value, location=None, overwrite=False)

property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.

class naqslab_devices.SignalGenerator.Models.HP_8648A (name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias

    description = 'HP 8648A Signal Generator'
    scale_factor = 1000000.0
    freq_limits = (100000.0, 1000000000.0)
    amp_scale_factor = 1.0
    amp_limits = (-136, 20)
    add_device (device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code (hdf5_file)
        Method to generate instructions for blacs_worker to program device.

        Must be over-ridden.

    get_all_children ()
    get_all_outputs ()
    get_properties (location=None)
        Get all properties in location

        If location is None we return all keys
    get_property (name, location=None, *args, **kwargs)
    init_device_group (hdf5_file)
    property parent_clock_line
    property pseudoclock_device
    quantise_amp (data, device)
        Quantize the amplitude in units of dBm and check it's within bounds
    quantise_freq (data, device)
        Quantize the frequency in units of Hz and check it's within bounds
    quantise_to_pseudoclock (times)
    set_properties (properties_dict, property_names, overwrite=False)
        Add one or a bunch of properties packed into properties_dict

```

property_names is a dictionary {key:val, ...} where each **val** is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (name, value, location=None, overwrite=False)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

class naqslab_devices.SignalGenerator.Models.HP_8648B (name, VISA_name)

Bases: *naqslab_devices.SignalGenerator.labscript_device.SignalGenerator*

VISA_name can be full VISA connection string or NI-MAX alias

description = 'HP 8648B Signal Generator'

scale_factor = 1000000.0

freq_limits = (9000.0, 2000000000.0)

amp_scale_factor = 1.0

amp_limits = (-136, 20)

add_device (device)

allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]

generate_code (hdf5_file)

Method to generate instructions for blacs_worker to program device.

Must be over-ridden.

get_all_children ()

get_all_outputs ()

get_properties (location=None)

Get all properties in location

If location is None we return all keys

get_property (name, location=None, *args, **kwargs)

init_device_group (hdf5_file)

property parent_clock_line

property pseudoclock_device

quantise_amp (data, device)

Quantize the amplitude in units of dBm and check it's within bounds

quantise_freq (data, device)

Quantize the frequency in units of Hz and check it's within bounds

quantise_to_pseudoclock (times)

set_properties (properties_dict, property_names, overwrite=False)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val, ...} where each **val** is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (name, value, location=None, overwrite=False)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

```

class naqslab_devices.SignalGenerator.Models.HP_8648C(name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'HP 8648C Signal Generator'
    scale_factor = 1000000.0
    freq_limits = (9000.0, 3200000000.0)
    amp_scale_factor = 1.0
    amp_limits = (-136, 20)
    add_device(device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code(hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.
    get_all_children()
    get_all_outputs()
    get_properties(location=None)
        Get all properties in location
        If location is None we return all keys
    get_property(name, location=None, *args, **kwargs)
    init_device_group(hdf5_file)
    property parent_clock_line
    property pseudoclock_device
    quantise_amp(data, device)
        Quantize the amplitude in units of dBm and check it's within bounds
    quantise_freq(data, device)
        Quantize the frequency in units of Hz and check it's within bounds
    quantise_to_pseudoclock(times)
    set_properties(properties_dict, property_names, overwrite=False)
        Add one or a bunch of properties packed into properties_dict
        property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
        to be pulled from properties_dict and added to the property with name key (it's location)
    set_property(name, value, location=None, overwrite=False)
    property t0
        The earliest time output can be commanded from this device at the start of the experiment. This is
        nonzero on secondary pseudoclock devices due to triggering delays.

class naqslab_devices.SignalGenerator.Models.HP_8648D(name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'HP 8648D Signal Generator'
    scale_factor = 1000000.0
    freq_limits = (9000.0, 4000000000.0)
    amp_scale_factor = 1.0

```

```
amp_limits = (-136, 20)
add_device(device)
allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
generate_code(hdf5_file)
    Method to generate instructions for blacs_worker to program device.
    Must be over-ridden.
get_all_children()
get_all_outputs()
get_properties(location=None)
    Get all properties in location
    If location is None we return all keys
get_property(name, location=None, *args, **kwargs)
init_device_group(hdf5_file)
property parent_clock_line
property pseudoclock_device
quantise_amp(data, device)
    Quantize the amplitude in units of dBm and check it's within bounds
quantise_freq(data, device)
    Quantize the frequency in units of Hz and check it's within bounds
quantise_to_pseudoclock(times)
set_properties(properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict
    property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)
set_property(name, value, location=None, overwrite=False)
property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.
class naqslab_devices.SignalGenerator.Models.E8257N(name, VISA_name)
    Bases: naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
    VISA_name can be full VISA connection string or NI-MAX alias
    description = 'Keysight E8257N Signal Generator'
    scale_factor = 1000000000.0
    freq_limits = (10000000.0, 40000000000.0)
    amp_scale_factor = 1.0
    amp_limits = (-105, 20)
    add_device(device)
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    generate_code(hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.
    get_all_children()
```



```

get_all_outputs ()
get_properties (location=None)
    Get all properties in location

    If location is None we return all keys
get_property (name, location=None, *args, **kwargs)
init_device_group (hdf5_file)
property parent_clock_line
property pseudoclock_device
quantise_amp (data, device)
    Quantize the amplitude in units of dBm and check it's within bounds
quantise_freq (data, device)
    Quantize the frequency in units of Hz and check it's within bounds
quantise_to_pseudoclock (times)
set_properties (properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict

    property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)
set_property (name, value, location=None, overwrite=False)
property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.
class naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.VISA.blacs_tab.VISATab

    You MUST override this method in order to define the device worker. You then call this parent method to
    finish initialization.

    base_units = {'amp': 'dBm', 'freq': 'MHz'}
    base_min = {'amp': -140, 'freq': 0.1}
    base_max = {'amp': 20, 'freq': 1057.5}
    base_step = {'amp': 0.1, 'freq': 1}
    base_decimals = {'amp': 1, 'freq': 6}
    status_byte_labels = {'bit 0': 'bit 0 label', 'bit 1': 'bit 1 label', 'bit 2': 'bit 2 label'}
    initialise_GUI ()
        Loads the standard STBstatus.ui widget and sets the worker defined in __init__
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    STBui_path = 'C:\\\\labscrip_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
    abort_buffered (*args, **kwargs)
    abort_transition_to_buffered (*args, **kwargs)
    add_secondary_worker (worker)
    auto_create_widgets ()

```

auto_place_widgets (**args*)

check_remote_values (**args, **kwargs*)

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)
Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call *finalise_close_tab()* to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)
Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)
Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()
Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

```

get_property(property)
get_save_data()
get_tab_layout()
hide_error()
initialise_workers()
mainloop()
property mode
on_force_full_buffered_reprogram()
on_resolve_value_inconsistency()
property primary_worker
program_device(*args, **kwargs)
program_device_properties(*args, **kwargs)
queue_work(worker_process, worker_function, *args, **kwargs)
restart(*args)
restore_built_in_save_data(data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data(data)
send_clear(*args, **kwargs)
set_tab_icon_and_colour()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible(visible)
shutdown_workers(*args, **kwargs)
start_run(*args, **kwargs)
property state
statemachine_timeout_add(delay, statefunction, *args, **kwargs)
statemachine_timeout_remove(statefunction)
statemachine_timeout_remove_all()
status_monitor(*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check(support)
supports_smart_programming(support)
transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)

class naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker(*args,
                                                                           **kwargs)
    Bases: naqslab_devices.VISA.blacs_worker.VISAWorker
    scale_factor = 1000000.0
    amp_scale_factor = 1.0
    freq_write_string = ''

```

freq_query_string = ''
freq_parser (*freq_string*)
Frequency Query string parser Needs to be over-ridden

amp_write_string = ''
amp_query_string = ''
amp_parser (*amp_string*)
Amplitude Query string parser Needs to be over-ridden

init ()
Initializes basic worker and opens VISA connection to device.
Default connection timeout is 2 seconds

check_remote_values ()

program_manual (*front_panel_values*)
Over-ride this method if remote programming is supported.
Returns `VISAWorker.check_remote_values()`

transition_to_buffered (*device_name, h5file, initial_values, fresh*)
Stores various device handles for use in `transition_to_manual` method.
Automatically called by BLACS. Should be over-ridden by inheritors.
Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

abort_buffered ()
Special abort shot code belongs here.

abort_transition_to_buffered ()
Special abort shot configuration code belongs here.

check_status ()
Reads the Status Byte Register of the VISA device.
Returns Status byte dictionary as formatted in `VISATab`
Return type dict

clear (*value*)
Sends standard *CLR to clear registers of device.
Parameters **value** (*bool*) – value of Clear button in `STBstatus.ui` widget

convert_register (*register*)
Converts returned register value to dict of bools
Parameters **register** (*int*) – Status register value returned from `read_stb`
Returns Status byte dictionary as formatted in `VISATab`
Return type dict

interrupt_startup (*reason='Process.interrupt_startup() called'*)
Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

```

mainloop ()

run (worker_name, device_name, extraargs)
    The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()
    Closes VISA connection to device.

start (*args, **kwargs)
    Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

terminate (wait_timeout=None, **kwargs)
    Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call Popen.terminate() and Popen.wait() on it.

transition_to_manual (abort=False)
    Simple transition_to_manual method where no data is saved.

class naqslab_devices.SignalGenerator.labscript_device.SignalGenerator (name,
                                                                    VISA_name)
    Bases: naqslab_devices.VISA.labscript_device.VISA
    VISA_name can be full VISA connection string or NI-MAX alias

    description = 'Signal Generator'
    allowed_children = [<class 'naqslab_devices.StaticFreqAmp'>]
    scale_factor = 1000000.0
    freq_limits = ()
    amp_scale_factor = 1.0
    amp_limits = ()

    quantise_freq (data, device)
        Quantize the frequency in units of Hz and check it's within bounds

    quantise_amp (data, device)
        Quantize the amplitude in units of dBm and check it's within bounds

    generate_code (hdf5_file)
        Method to generate instructions for blacs_worker to program device.
        Must be over-ridden.

    add_device (device)

    get_all_children ()

    get_all_outputs ()

    get_properties (location=None)
        Get all properties in location
        If location is None we return all keys

    get_property (name, location=None, *args, **kwargs)

    init_device_group (hdf5_file)

    property parent_clock_line

    property pseudoclock_device

    quantise_to_pseudoclock (times)

    set_properties (properties_dict, property_names, overwrite=False)
        Add one or a bunch of properties packed into properties_dict

```

property_names is a dictionary {key:val, ...} where each **val** is a list [var1, var2, ...] of variables to be pulled from **properties_dict** and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property **t0**

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

Configures which BLACS_tab goes to which labscript_device.

3.4 PulseBlasterESRPro300

3.4.1 Overview

This is a thin subclass of the PulseBlaster_No_DDS labscript_device. It merely configures the correct clock speed, digital output number, and clock resolution limits.

```
naqslab_devices.  
PulseBlasterESRPro300.blacs_tab
```

```
naqslab_devices.  
PulseBlasterESRPro300.blacs_worker
```

```
naqslab_devices.  
PulseBlasterESRPro300.  
labscript_device
```

```
naqslab_devices.  
PulseBlasterESRPro300.  
register_classes
```

```
naqslab_devices.  
PulseBlasterESRPro300.  
runviewer_parser
```

3.4.2 Detailed Documentation of naqslab_devices.PulseBlasterESRPro300

PulseBlasterESRPro300.blacs_tab

```
class naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab (*args,  
                                                                              **kwargs)  
    Bases: labscript_devices.PulseBlaster_No_DDS.Pulseblaster_No_DDS_Tab  
    num_DO = 24  
    ICON_BUSY = ':/qtutils/fugue/hourglass'  
    ICON_ERROR = ':/qtutils/fugue/exclamation'  
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'  
    ICON_OK = ':/qtutils/fugue/tick'  
    abort_buffered (*args, **kwargs)  
    abort_transition_to_buffered (*args, **kwargs)  
    add_secondary_worker (worker)  
    auto_create_widgets ()  
    auto_place_widgets (*args)  
    check_remote_values (*args, **kwargs)
```

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)
Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call *finalise_close_tab()* to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)
Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)
Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()
Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

```
get_tab_layout ()
hide_error ()
initialise_GUI ()
initialise_workers ()
labscript_device_class_name = 'PulseBlaster_No_DDS'
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
reset (*args, **kwargs)
restart (*args)
restore_builtin_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
status_monitor (*args, **kwargs)
stop (*args, **kwargs)
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)
```


PulseBlasterESRPro300.blacs_worker

```

class naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker (*args, **kwargs)
    Bases: labscript_devices.PulseBlaster_No_DDS.PulseblasterNoDDSWorker
    core_clock_freq = 300.0
    ESRPro = True
    abort_buffered()
    abort_transition_to_buffered()
    check_status()
    init()
    interrupt_startup (reason='Process.interrupt_startup() called')
        Called from the parent process. Interrupt all blocking operations on starting the child process, causing
        Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was
        interrupted before the child was started, otherwise self.child will be the child Popen object, which
        could be at any stage of setting up its connection with the parent. This method may be called multiple
        times without raising an exception, it will simply do nothing if startup has previously been interrupted
    mainloop()
    program_manual (values)
    run (worker_name, device_name, extraargs)
        The method that gets called in the subprocess. To be overridden by subclasses
    shutdown()
    start (*args, **kwargs)
        Call in the parent process to start a subprocess. Passes args and kwargs to the run() method
    start_run()
    terminate (wait_timeout=None, **kwargs)
        Interrupt process startup if not already done, ensuring self.child exists or is None if startup was in-
        terrupted before the process was created. Then if the child is not None, call Popen.terminate() and
        Popen.wait() on it.
    transition_to_buffered (device_name, h5file, initial_values, fresh)
    transition_to_manual()

```

PulseBlasterESRPro300.labscript_device

```

class naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300 (name,
trig-
ger_de
trig-
ger_co
board_
firmwa
pro-
gram-
ming_s
pulse_
max_in
time_b
time_b
**kwargs)
    Bases: labscript_devices.PulseBlaster_No_DDS.PulseBlaster_No_DDS

```

```
description = 'SpinCore PulseBlaster ESR-PRO-300'
clock_limit = 30000000.0
clock_resolution = 4e-09
n_flags = 24
add_device(device)
allowed_children = [<class 'labscript.labscript.Pseudoclock'>]
convert_to_pb_inst(dig_outputs, dds_outputs, freqs, amps, phases)
core_clock_freq = 100
property direct_outputs
do_checks(outputs)
    Basic error checking to ensure the user's instructions make sense
flag_is_clock(flag)
flag_valid(flag)
generate_code(hdf5_file)
generate_registers(hdf5_file, dds_outputs)
get_all_children()
get_all_outputs()
get_direct_outputs()
    Finds out which outputs are directly attached to the PulseBlaster
get_flag_number(connection)
get_properties(location=None)
    Get all properties in location
    If location is None we return all keys
get_property(name, location=None, *args, **kwargs)
init_device_group(hdf5_file)
property is_master_pseudoclock
minimum_recovery_time = 0
offset_instructions_from_trigger(outputs)
property parent_clock_line
pb_instructions = {'BRANCH': 6, 'CONTINUE': 0, 'END_LOOP': 3, 'LONG_DELAY': 7, 'LOOP'}
property pseudoclock
property pseudoclock_device
quantise_to_pseudoclock(times)
set_initial_trigger_time(t)
set_properties(properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict
    property_names is a dictionary {key:val,...} where each val is a list [var1, var2,...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)
set_property(name, value, location=None, overwrite=False)
```

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

trigger (*t, duration, wait_delay=0*)

Ask the trigger device to produce a digital pulse of a given duration to trigger this pseudoclock

trigger_delay = 2.5e-07**trigger_edge_type** = 'falling'**trigger_minimum_duration** = 0**wait_delay** = 1e-07**write_pb_inst_to_h5** (*pb_inst, hdf5_file*)**PulseBlasterESRPro300.register_classes****PulseBlasterESRPro300.runviewer_parser**

```
class naqslab_devices.PulseBlasterESRPro300.runviewer_parser.PulseBlasterESRPro300Parser
```

```
Bases: labscript_devices.PulseBlaster_No_DDS.PulseBlaster_No_DDS_Parser
```

```
num_flags = 24
```

```
get_traces (add_trace, parent=None)
```

```
labscript_device_class_name = 'PulseBlaster_No_DDS'
```

```
num_dds = 0
```

3.5 PulseBlaster_No_DDS_200

3.5.1 Overview

This is a thin subclass of the PulseBlaster_No_DDS labscript_device. It merely configures the correct clock speed and clock resolution limits for our custom 200 MHz clocked USB PulseBlaster board.

```
naqslab_devices.  
PulseBlaster_No_DDS_200.blacs_tab
```

```
naqslab_devices.  
PulseBlaster_No_DDS_200.  
blacs_worker
```

```
naqslab_devices.  
PulseBlaster_No_DDS_200.  
labscript_device
```

```
naqslab_devices.  
PulseBlaster_No_DDS_200.  
register_classes
```

```
naqslab_devices.  
PulseBlaster_No_DDS_200.  
runviewer_parser
```

3.5.2 Detailed Documentation of naqslab_devices.PulseBlaster_No_DDS_200

PulseBlaster_No_DDS_200.blacs_tab

```
class naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab(*args, **kwargs)
    Bases: labscript_devices.PulseBlaster_No_DDS.Pulseblaster_No_DDS_Tab
    num_DO = 24
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    abort_buffered(*args, **kwargs)
    abort_transition_to_buffered(*args, **kwargs)
    add_secondary_worker(worker)
    auto_create_widgets()
    auto_place_widgets(*args)
    check_remote_values(*args, **kwargs)
    check_time()
    clean_ui_on_restart()
    close_tab(finalise=True)
        Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
    connect_restart_receiver(function)
    continue_restart(currentpage)
        Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
    create_analog_outputs(analog_properties)
    create_analog_widgets(channel_properties)
    create_dds_outputs(dds_properties)
    create_dds_widgets(channel_properties)
    create_device_properties(device_properties)
    create_digital_outputs(digital_properties)
    create_digital_widgets(channel_properties)
    create_image_outputs(image_properties)
    create_image_widgets(channel_properties)
    create_property_widgets(device_properties)
    create_worker(name, WorkerClass, workerargs=None)
        Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.
```

```

property device_name
disconnect_restart_receiver (function)
property error_message
finalise_close_tab (currentpage)
finalise_restart (currentpage)
property force_full_buffered_reprogram
get_all_save_data ()
get_builtin_save_data ()
    Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.
get_channel (channel)
get_child_from_connection_table (parent_device_name, port)
get_front_panel_properties ()
get_front_panel_values ()
get_property (property)
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_GUI ()
initialise_workers ()
labscript_device_class_name = 'PulseBlaster_No_DDS'
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
reset (*args, **kwargs)
restart (*args)
restore_built_in_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start (*args, **kwargs)
start_run (*args, **kwargs)

```

```
property state
statemachine_timeout_add(delay, statefunction, *args, **kwargs)
statemachine_timeout_remove(statefunction)
statemachine_timeout_remove_all()
status_monitor(*args, **kwargs)
stop(*args, **kwargs)
supports_remote_value_check(support)
supports_smart_programming(support)
transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)
```

PulseBlaster_No_DDS_200.blacs_worker

```
class naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseblasterNoDDS200Worker(*a
**a

    Bases: labscript_devices.PulseBlaster_No_DDS.PulseblasterNoDDSWorker

    core_clock_freq = 200.0

    abort_buffered()

    abort_transition_to_buffered()

    check_status()

    init()

    interrupt_startup(reason='Process.interrupt_startup() called')
        Called from the parent process. Interrupt all blocking operations on starting the child process, causing
        Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was
        interrupted before the child was started, otherwise self.child will be the child Popen object, which
        could be at any stage of setting up its connection with the parent. This method may be called multiple
        times without raising an exception, it will simply do nothing if startup has previously been interrupted

    mainloop()

    program_manual(values)

    run(worker_name, device_name, extraargs)
        The method that gets called in the subprocess. To be overridden by subclasses

    shutdown()

    start(*args, **kwargs)
        Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

    start_run()

    terminate(wait_timeout=None, **kwargs)
        Interrupt process startup if not already done, ensuring self.child exists or is None if startup was in-
        terrupted before the process was created. Then if the child is not None, call Popen.terminate() and
        Popen.wait() on it.

    transition_to_buffered(device_name, h5file, initial_values, fresh)

    transition_to_manual()
```

PulseBlaster_No_DDS_200.labscript_device

```
class naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200 (n
```

Bases: labscript_devices.PulseBlaster_No_DDS.PulseBlaster_No_DDS

A thin subclass of labscript_devices.PulseBlaster_No_DDS.

It's only purpose is to set the core clock frequency to 200 MHz for our one custom USB pulseblaster device.

```
description = 'SpinCore PulseBlaster USB with 200 MHz clock'
clock_limit = 17200000.0
clock_resolution = 1e-08
core_clock_freq = 200
n_flags = 24
add_device (device)
allowed_children = [<class 'labscript.labscript.Pseudoclock'>]
convert_to_pb_inst (dig_outputs, dds_outputs, freqs, amps, phases)
property direct_outputs
do_checks (outputs)
    Basic error checking to ensure the user's instructions make sense
flag_is_clock (flag)
flag_valid (flag)
generate_code (hdf5_file)
generate_registers (hdf5_file, dds_outputs)
get_all_children ()
get_all_outputs ()
get_direct_outputs ()
    Finds out which outputs are directly attached to the PulseBlaster
get_flag_number (connection)
get_properties (location=None)
    Get all properties in location
    If location is None we return all keys
get_property (name, location=None, *args, **kwargs)
init_device_group (hdf5_file)
```

```
property is_master_pseudoclock
minimum_recovery_time = 0
offset_instructions_from_trigger (outputs)
property parent_clock_line
pb_instructions = {'BRANCH': 6, 'CONTINUE': 0, 'END_LOOP': 3, 'LONG_DELAY': 7, 'LOOP'
property pseudoclock
property pseudoclock_device
quantise_to_pseudoclock (times)
set_initial_trigger_time (t)
set_properties (properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict
    property_names is a dictionary {key:val,...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)
set_property (name, value, location=None, overwrite=False)
property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.
trigger (t, duration, wait_delay=0)
    Ask the trigger device to produce a digital pulse of a given duration to trigger this pseudoclock
trigger_delay = 2.5e-07
trigger_edge_type = 'falling'
trigger_minimum_duration = 0
wait_delay = 1e-07
write_pb_inst_to_h5 (pb_inst, hdf5_file)
```

PulseBlaster_No_DDS_200.register_classes

PulseBlaster_No_DDS_200.runviewer_parser

```
class naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser.PulseBlaster_No_DDS_200_Parser

    Bases: labscript_devices.PulseBlaster_No_DDS.PulseBlaster_No_DDS_Parser
    num_flags = 24
    get_traces (add_trace, parent=None)
    labscript_device_class_name = 'PulseBlaster_No_DDS'
    num_dds = 0
```

3.6 KeysightXSeries

3.6.1 Overview

This devices covers Keysight X-series oscilloscopes. It has been explicitly tested with 1000 and 3000 series 'scopes. Other series should be simple to implement.

```

naqslab_devices.KeysightXSeries.
blacs_tab
naqslab_devices.KeysightXSeries.
blacs_worker
naqslab_devices.KeysightXSeries.
labscript_device
naqslab_devices.KeysightXSeries.
register_classes

```

3.6.2 Detailed Documentation of naqslab_devices.KeysightXSeries

KeysightXSeries.blacs_tab

class naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab (*args,
**kwargs)

Bases: *naqslab_devices.VISA.blacs_tab.VISATab*

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

```

status_byte_labels = {'bit 0': 'Operation Complete', 'bit 1': 'Unused', 'bit 2':
initialise_GUI ()
    Loads the standard STBstatus.ui widget and sets the worker defined in __init__
ICON_BUSY = ':/qtutils/fugue/hourglass'
ICON_ERROR = ':/qtutils/fugue/exclamation'
ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
ICON_OK = ':/qtutils/fugue/tick'
STBui_path = 'C:\\labscript_suite\\naqslab_devices\\VISA\\STBstatus.ui'
abort_buffered (*args, **kwargs)
abort_transition_to_buffered (*args, **kwargs)
add_secondary_worker (worker)
auto_create_widgets ()
auto_place_widgets (*args)
check_remote_values (*args, **kwargs)
check_time ()
clean_ui_on_restart ()
close_tab (finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver (function)
continue_restart (currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread.
    Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs (analog_properties)
create_analog_widgets (channel_properties)
create_dds_outputs (dds_properties)

```

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name*, *WorkerClass*, *workerargs=None*)
Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()
Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (**args*, ***kwargs*)

program_device_properties (**args*, ***kwargs*)

queue_work (*worker_process*, *worker_function*, **args*, ***kwargs*)

restart (**args*)

restore_builtin_save_data (*data*)

Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (*data*)

send_clear (**args, **kwargs*)

set_tab_icon_and_colour ()

Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour respectively

set_terminal_visible (*visible*)

shutdown_workers (**args, **kwargs*)

start_run (**args, **kwargs*)

property state

statemachine_timeout_add (*delay, statefunction, *args, **kwargs*)

statemachine_timeout_remove (*statefunction*)

statemachine_timeout_remove_all ()

status_monitor (**args, **kwargs*)

status_widget = 'STBstatus.ui'

supports_remote_value_check (*support*)

supports_smart_programming (*support*)

transition_to_buffered (**args, **kwargs*)

transition_to_manual (**args, **kwargs*)

update_from_settings (*settings*)

KeysightXSeries.blacs_worker

class naqslab_devices.KeysightXSeries.blacs_worker.**KeysightXScopeWorker** (**args, **kwargs*)

Bases: *naqslab_devices.VISA.blacs_worker.VISAWorker*

setup_string = '*ESE 60;*SRE 32;*CLS;:WAV:BYT MSBF;UNS ON;POIN:MODE RAW'

esr_mask = 60

read_analog_parameters_string = ':WAV:FORM WORD;SOUR CHAN{0:d};PRE?'

read_dig_parameters_string = ':WAV:FORM BYTE;SOUR POD{0:d};PRE?'

read_waveform_string = ':WAV:DATA?'

read_counter_string = ':MEAS:{0:s}{1:s}? CHAN{2:d}'

model_ident = 'SO-X'

dig_command = ':DIG'

analog_waveform_parser (*raw_waveform_array, y0, dy, yoffset*)

Parses the numpy array from the analog waveform query.

digital_pod_parser (*raw_pod_array*)

Unpacks the bits for a pod array Columns returned are in bit order [7,6,5,4,3,2,1,0]

error_parser (*error_return_string*)

Parses the strings returned by :SYST:ERR? Returns int_code, err_string

init()
Initializes basic worker and opens VISA connection to device.
Default connection timeout is 2 seconds

transition_to_buffered(*device_name, h5file, initial_values, fresh*)
This configures counters, if any are defined, as well as optional compression options for saved data traces.

transition_to_manual(*abort=False*)
Simple transition_to_manual method where no data is saved.

check_status()
Periodically called by BLACS to check to status of the scope.

abort_buffered()
Special abort shot code belongs here.

abort_transition_to_buffered()
Special abort shot configuration code belongs here.

check_remote_values()

clear(*value*)
Sends standard *CLR to clear registers of device.
Parameters *value* (*bool*) – value of Clear button in STBstatus.ui widget

convert_register(*register*)
Converts returned register value to dict of bools
Parameters *register* (*int*) – Status register value returned from `read_stb`
Returns Status byte dictionary as formatted in VISATab
Return type dict

interrupt_startup(*reason='Process.interrupt_startup() called'*)
Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop()

program_manual(*front_panel_values*)
Over-ride this method if remote programming is supported.
Returns `VISAWorker.check_remote_values()`

run(*worker_name, device_name, extraargs*)
The method that gets called in the subprocess. To be overridden by subclasses

shutdown()
Closes VISA connection to device.

start(**args, **kwargs*)
Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate(*wait_timeout=None, **kwargs*)
Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

KeysightXSeries.labscript_device

```
class naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope (name,
                                                                    VISA_name,
                                                                    trig-
                                                                    ger_device,
                                                                    trig-
                                                                    ger_connection,
                                                                    num_AI=4,
                                                                    DI=True,
                                                                    trig-
                                                                    ger_duration=0.001,
                                                                    com-
                                                                    pres-
                                                                    sion=None,
                                                                    com-
                                                                    pres-
                                                                    sion_opts=None,
                                                                    shuf-
                                                                    fle=False,
                                                                    **kwargs)
```

Bases: `labscript.labscript.TriggerableDevice`

VISA_name can be full VISA connection string or NI-MAX alias. Trigger Device should be fast clocked device. num_AI sets number of analog input channels, default 4 DI sets if DI are present, default True trigger_duration set scope trigger duration, default 1ms Compression of traces in h5 file controlled by: compression: 'lzf', 'gzip', None compression_opts: 0-9 for gzip shuffle: True/False

description = 'Keysight X Series Digital Oscilloscope'

allowed_children = [`<class 'naqslab_devices.ScopeChannel'>`]

generate_code (*hdf5_file*)

Automatically called by compiler to write acquisition instructions to h5 file. Configures counters, analog and digital acquisitions.

acquire (*start_time*)

Call to define time when trigger will happen for scope.

add_device (*device*)

do_checks ()

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

minimum_recovery_time = 0

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val, ...} where each **val** is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

trigger (*t, duration*)

Request parent trigger device to produce a trigger at time t with given duration.

trigger_edge_type = 'rising'

KeysightXSeries.register_classes

3.7 NovaTechDDS

3.7.1 Overview

This module covers the 409B-AC, 409B, and 440A DDS frequency synthesizers from Novatech.

Note that the 409B-AC class is largely interchangeable with the DDS9m labscript_devices class.

naqslab_devices.NovaTechDDS.

blacs_tab

naqslab_devices.NovaTechDDS.

blacs_worker

naqslab_devices.NovaTechDDS.

labscript_device

naqslab_devices.NovaTechDDS.

register_classes

naqslab_devices.NovaTechDDS.

runviewer_parser

3.7.2 Detailed Documentation of naqslab_devices.NovaTechDDS

NovaTechDDS.blacs_tab

```
class naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab (*args,
                                                                **kwargs)
    Bases: blacs.device_base_class.DeviceTab
    initialise_GUI()
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    abort_buffered(*args, **kwargs)
    abort_transition_to_buffered(*args, **kwargs)
    add_secondary_worker(worker)
    auto_create_widgets()
    auto_place_widgets(*args)
```

check_remote_values (**args, **kwargs*)

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)
 Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call *finalise_close_tab()* to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)
 Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)
 Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (*function*)

property error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

property force_full_buffered_reprogram

get_all_save_data ()

get_builtin_save_data ()
 Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name, port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

```
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_builtin_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)
class naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab (*args,
                                                            **kwargs)
    Bases: naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
    ICON_OK = ':/qtutils/fugue/tick'
    abort_buffered (*args, **kwargs)
    abort_transition_to_buffered (*args, **kwargs)
    add_secondary_worker (worker)
```


auto_create_widgets ()

auto_place_widgets (*args)

check_remote_values (*args, **kwargs)

check_time ()

clean_ui_on_restart ()

close_tab (finalise=True)
 Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations

connect_restart_receiver (function)

continue_restart (currentpage)
 Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (analog_properties)

create_analog_widgets (channel_properties)

create_dds_outputs (dds_properties)

create_dds_widgets (channel_properties)

create_device_properties (device_properties)

create_digital_outputs (digital_properties)

create_digital_widgets (channel_properties)

create_image_outputs (image_properties)

create_image_widgets (channel_properties)

create_property_widgets (device_properties)

create_worker (name, WorkerClass, workerargs=None)
 Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property device_name

disconnect_restart_receiver (function)

property error_message

finalise_close_tab (currentpage)

finalise_restart (currentpage)

property force_full_buffered_reprogram

get_all_save_data ()

get_built_in_save_data ()
 Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (channel)

get_child_from_connection_table (parent_device_name, port)

get_front_panel_properties ()

```
get_front_panel_values ()
get_property (property)
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_GUI ()
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_built_in_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)
class naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab (*args,
                                                            **kwargs)
    Bases: naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
    initialise_GUI ()
    ICON_BUSY = ':/qtutils/fugue/hourglass'
    ICON_ERROR = ':/qtutils/fugue/exclamation'
    ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
```

```

ICON_OK = ':/qtutils/fugue/tick'
abort_buffered (*args, **kwargs)
abort_transition_to_buffered (*args, **kwargs)
add_secondary_worker (worker)
auto_create_widgets ()
auto_place_widgets (*args)
check_remote_values (*args, **kwargs)
check_time ()
clean_ui_on_restart ()
close_tab (finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver (function)
continue_restart (currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs (analog_properties)
create_analog_widgets (channel_properties)
create_dds_outputs (dds_properties)
create_dds_widgets (channel_properties)
create_device_properties (device_properties)
create_digital_outputs (digital_properties)
create_digital_widgets (channel_properties)
create_image_outputs (image_properties)
create_image_widgets (channel_properties)
create_property_widgets (device_properties)
create_worker (name, WorkerClass, workerargs=None)
    Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.
property device_name
disconnect_restart_receiver (function)
property error_message
finalise_close_tab (currentpage)
finalise_restart (currentpage)
property force_full_buffered_reprogram
get_all_save_data ()

```

get_builtin_save_data ()
Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.

get_channel (*channel*)

get_child_from_connection_table (*parent_device_name*, *port*)

get_front_panel_properties ()

get_front_panel_values ()

get_property (*property*)

get_save_data ()

get_tab_layout ()

hide_error ()

initialise_workers ()

mainloop ()

property mode

on_force_full_buffered_reprogram ()

on_resolve_value_inconsistency ()

property primary_worker

program_device (**args*, ***kwargs*)

program_device_properties (**args*, ***kwargs*)

queue_work (*worker_process*, *worker_function*, **args*, ***kwargs*)

restart (**args*)

restore_built_in_save_data (*data*)
Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.

restore_save_data (*data*)

set_tab_icon_and_colour ()
Set the tab icon and the colour of its text to the values of *self._tab_icon* and *self._tab_text_colour* respectively

set_terminal_visible (*visible*)

shutdown_workers (**args*, ***kwargs*)

start_run (**args*, ***kwargs*)

property state

statemachine_timeout_add (*delay*, *statefunction*, **args*, ***kwargs*)

statemachine_timeout_remove (*statefunction*)

statemachine_timeout_remove_all ()

supports_remote_value_check (*support*)

supports_smart_programming (*support*)

transition_to_buffered (**args*, ***kwargs*)

transition_to_manual (**args*, ***kwargs*)

update_from_settings (*settings*)

NovaTechDDS.blacs_worker

```

class naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker (*args,
                                                                    **kwargs)
    Bases: blacs.tab_base_classes.Worker

    init ()
        Initialization command run automatically by the BLACS tab on startup. It establishes communication
        and sends initial default configuration commands

    check_connection ()
        Sends non-command and tests for correct response returns tuple of connection state and response string

    write_check (command)
        Sends command and checks and confirms proper execution by reading 'OK' from device.

    check_error (response)
        Parse response for errors and raise appropriate error. If no error, returns response unaltered.

    check_remote_values ()
        Queries device for current output settings. Return results as a dictionary to update the BLACS tab.

    program_manual (front_panel_values)
        Called within the BLACS worker during transitions. This calls program_static for each setting if it
        isn't already set.

    program_static (channel, type, value)
        General output parameter programming function. Only sends one command per use.

    transition_to_buffered (device_name, h5file, initial_values, fresh)

    abort_transition_to_buffered ()

    abort_buffered ()

    transition_to_manual (abort=False)

    shutdown ()

    interrupt_startup (reason='Process.interrupt_startup() called')
        Called from the parent process. Interrupt all blocking operations on starting the child process, causing
        Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was
        interrupted before the child was started, otherwise self.child will be the child Popen object, which
        could be at any stage of setting up its connection with the parent. This method may be called multiple
        times without raising an exception, it will simply do nothing if startup has previously been interrupted

    mainloop ()

    run (worker_name, device_name, extraargs)
        The method that gets called in the subprocess. To be overridden by subclasses

    start (*args, **kwargs)
        Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

    terminate (wait_timeout=None, **kwargs)
        Interrupt process startup if not already done, ensuring self.child exists or is None if startup was in-
        terrupted before the process was created. Then if the child is not None, call Popen.terminate() and
        Popen.wait() on it.

class naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker (*args,
                                                                    **kwargs)
    Bases: naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker

    transition_to_manual (abort=False)

    abort_buffered ()

    abort_transition_to_buffered ()

```

check_connection ()
Sends non-command and tests for correct response returns tuple of connection state and reponse string

check_error (*response*)
Parse response for errors and raise appropriate error. If no error, returns response unaltered.

check_remote_values ()
Queries device for current output settings. Return results as a dictionary to update the BLACS tab.

init ()
Initialization command run automatically by the BLACS tab on startup. It establishes communication and sends initial default configuration commands

interrupt_startup (*reason='Process.interrupt_startup() called'*)
Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

program_manual (*front_panel_values*)
Called within the BLACS worker during transitions. This calls `program_static` for each setting if it isn't already set.

program_static (*channel, type, value*)
General output parameter programming function. Only sends one command per use.

run (*worker_name, device_name, extraargs*)
The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

start (**args, **kwargs*)
Call in the parent process to start a subprocess. Passes `args` and `kwargs` to the `run()` method

terminate (*wait_timeout=None, **kwargs*)
Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

write_check (*command*)
Sends command and checks and confirms proper execution by reading 'OK' from device.

class `naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker` (**args, **kwargs*)
Bases: `naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`

init ()
Modified init from 409B-AC. The 440A only supports one baud rate and does not support output mode commands.

program_static (*channel, type, value*)
General output parameter programming function. Only sends one command per use.

check_remote_values ()
The 440A Query command returns values in a different order and does not tell the amplitude.

abort_buffered ()

abort_transition_to_buffered ()

check_connection ()
Sends non-command and tests for correct response returns tuple of connection state and reponse string

check_error (*response*)

Parse response for errors and raise appropriate error. If no error, returns response unaltered.

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was interrupted before the child was started, otherwise self.child will be the child Popen object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

program_manual (*front_panel_values*)

Called within the BLACS worker during transitions. This calls program_static for each setting if it isn't already set.

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call Popen.terminate() and Popen.wait() on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

transition_to_manual (*abort=False*)

write_check (*command*)

Sends command and checks and confirms proper execution by reading 'OK' from device.

NovaTechDDS.labscript_device

```
class naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC (name,
                                                                    par-
                                                                    ent_device,
                                                                    com_port=",",
                                                                    baud_rate=19200,
                                                                    up-
                                                                    date_mode='synchronous',
                                                                    syn-
                                                                    chronous_first_line_repeat=False,
                                                                    phase_mode='continuous',
                                                                    ext_clk=False,
                                                                    clk_freq=None,
                                                                    clk_mult=None,
                                                                    R_option=False,
                                                                    **kwargs)
```

Bases: labscript.labscript.IntermediateDevice

Labscript device class for NovaTech 409B-AC variant DDS. This device has two dynamic channels (0,1) and two static channels (2,3). If an external clock frequency is enabled, and /R option is not being used, clk_freq (in MHz) and clk_mult (int) must also be defined.

description = 'NT-DDS409B-AC'

allowed_children = [<class 'labscript.labscript.DDS'>, <class 'labscript.labscript.DDS'>]

clock_limit = 9990

clock_check ()

Checks to make sure `clk_mult` and `clk_freq` have valid values, as determined by the Novatech documentation. Returns the correct frequency scaling factor to account for different clocking options.

add_device (*device*)

get_default_unit_conversion_classes (*device*)

Child devices call this during their `__init__` (with themselves as the argument) to check if there are certain unit calibration classes that they should apply to their outputs, if the user has not otherwise specified a calibration class

quantise_freq (*data, device*)

Provides bounds error checking and scales input values to instrument units (0.1 Hz) before ensuring uint32 integer type.

quantise_phase (*data, device*)

Ensures phase is wrapped about 360 degrees and scales to instrument units before type casting to uint16.

quantise_amp (*data, device*)

Ensures amplitude is within bounds and scales to instrument units (between 0 and 1023) before type-casting to uint16

generate_code (*hdf5_file*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into `properties_dict`

property_names is a dictionary **{key:val, ...}** where each **val** is a list [`var1`, `var2`, ...] of variables to be pulled from `properties_dict` and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

```
class naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B (name,  
                                                                com_port=,"  
                                                                baud_rate=19200,  
                                                                phase_mode='default',  
                                                                R_option=False,  
                                                                ext_clk=False,  
                                                                clk_freq=None,  
                                                                clk_mult=None,  
                                                                **kwargs)
```

Bases: `naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC`

Labscript class for NovaTech 409B DDS. This device has four static DDS output channels.


```

description = 'NT-DDS409B'

allowed_children = [<class 'labscript.labscript.StaticDDS'>]

clock_limit = 1

generate_code(hdf5_file)
    Modified version of 409B-AC generate_code that only handles static DDS outputs

add_device(device)

clock_check()
    Checks to make sure clk_mult and clk_freq have valid values, as determined by the Novatech docu-
    mentation. Returns the correct frequency scaling factor to account for different clocking options.

get_all_children()

get_all_outputs()

get_default_unit_conversion_classes(device)
    Child devices call this during their __init__ (with themselves as the argument) to check if there are
    certain unit calibration classes that they should apply to their outputs, if the user has not otherwise
    specified a calibration class

get_properties(location=None)
    Get all properties in location

    If location is None we return all keys

get_property(name, location=None, *args, **kwargs)

init_device_group(hdf5_file)

property parent_clock_line

property pseudoclock_device

quantise_amp(data, device)
    Ensures amplitude is within bounds and scales to instrument units (between 0 and 1023) before type-
    casting to uint16

quantise_freq(data, device)
    Provides bounds error checking and scales input values to instrument units (0.1 Hz) before ensuring
    uint32 integer type.

quantise_phase(data, device)
    Ensures phase is wrapped about 360 degrees and scales to instrument units before type casting to
    uint16.

quantise_to_pseudoclock(times)

set_properties(properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict

    property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)

set_property(name, value, location=None, overwrite=False)

property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.

class naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A(name,
                                                                    com_port="",
                                                                    baud_rate=19200,
                                                                    ext_clk=False,
                                                                    clk_freq=None,
                                                                    **kwargs)
    Bases: naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC

```

Labscript class for Novatech 440A DDS. This is a high frequency DDS with single channel output that does not support amplitude control

description = 'NT-DDS440A'

allowed_children = [<class 'labscript.labscript.StaticDDS'>]

clock_limit = 1

add_device (*device*)

quantise_freq (*data, device*)

Provides bounds error checking and scales input values to instrument units (0.1 Hz) before ensuring uint32 integer type.

generate_code (*hdf5_file*)

Modified generate code from 409B to only accept one channel and ignore amplitude commands which are not supported by the device.

clock_check ()

Checks to make sure clk_mult and clk_freq have valid values, as determined by the Novatech documentation. Returns the correct frequency scaling factor to account for different clocking options.

get_all_children ()

get_all_outputs ()

get_default_unit_conversion_classes (*device*)

Child devices call this during their __init__ (with themselves as the argument) to check if there are certain unit calibration classes that they should apply to their outputs, if the user has not otherwise specified a calibration class

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_amp (*data, device*)

Ensures amplitude is within bounds and scales to instrument units (between 0 and 1023) before type-casting to uint16

quantise_phase (*data, device*)

Ensures phase is wrapped about 360 degrees and scales to instrument units before type casting to uint16.

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

NovaTechDDS.register_classes

NovaTechDDS.runviewer_parser

```
class naqslab_devices.NovaTechDDS.runviewer_parser.NovaTech409B_ACParser (path,
                                                                    de-
                                                                    vice)
```

Bases: object

get_traces (add_trace, clock=None)

```
class naqslab_devices.NovaTechDDS.runviewer_parser.NovaTech409BParser (path,
                                                                    de-
                                                                    vice)
```

Bases: *naqslab_devices.NovaTechDDS.runviewer_parser.NovaTech409B_ACParser*

get_traces (add_trace, clock=None)

```
class naqslab_devices.NovaTechDDS.runviewer_parser.NovaTech440AParser (path,
                                                                    de-
                                                                    vice)
```

Bases: *naqslab_devices.NovaTechDDS.runviewer_parser.NovaTech409B_ACParser*

get_traces (add_trace, clock=None)

3.8 SR865

3.8.1 Overview

This device class controls the Stanford Research Systems 865 series Lock-in Amplifier. It only implements control functions. Data acquisition is not directly supported at this time. To get buffered data, please use a DAQ connected to the analog outputs of the 865.

```
naqslab_devices.SR865.blacs_tab
naqslab_devices.SR865.blacs_worker
naqslab_devices.SR865.
labscript_device
naqslab_devices.SR865.
register_classes
```

3.8.2 Detailed Documentation of naqslab_devices.SR865

SR865.blacs_tab

```
class naqslab_devices.SR865.blacs_tab.SR865Tab (*args, **kwargs)
```

Bases: *naqslab_devices.VISA.blacs_tab.VISATab*

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

```
status_byte_labels = {'bit 0': 'OPC', 'bit 1': 'Input Queue Overflow', 'bit 2':
```

```
initialise_GUI ()
```

Loads the standard STBstatus.ui widget and sets the worker defined in __init__

```
tau_changed (*args, **kwargs)
```

```
sens_changed (*args, **kwargs)
```

```
ICON_BUSY = ':/qtutils/fugue/hourglass'
```

ICON_ERROR = `':/qtutils/fugue/exclamation'`

ICON_FATAL_ERROR = `':/qtutils/fugue/exclamation-red'`

ICON_OK = `':/qtutils/fugue/tick'`

STBui_path = `'C:\\labscript_suite\\naqslab_devices\\VISA\\STBstatus.ui'`

abort_buffered (**args, **kwargs*)

abort_transition_to_buffered (**args, **kwargs*)

add_secondary_worker (*worker*)

auto_create_widgets ()

auto_place_widgets (**args*)

check_remote_values (**args, **kwargs*)

check_time ()

clean_ui_on_restart ()

close_tab (*finalise=True*)
Close the tab, terminate subprocesses and join the mainloop thread. If *finalise=False*, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call `finalise_close_tab()` to perform these potentially blocking operations

connect_restart_receiver (*function*)

continue_restart (*currentpage*)
Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.

create_analog_outputs (*analog_properties*)

create_analog_widgets (*channel_properties*)

create_dds_outputs (*dds_properties*)

create_dds_widgets (*channel_properties*)

create_device_properties (*device_properties*)

create_digital_outputs (*digital_properties*)

create_digital_widgets (*channel_properties*)

create_image_outputs (*image_properties*)

create_image_widgets (*channel_properties*)

create_property_widgets (*device_properties*)

create_worker (*name, WorkerClass, workerargs=None*)
Set up a worker process. *WorkerClass* can either be a subclass of *Worker*, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.

property_device_name

disconnect_restart_receiver (*function*)

property_error_message

finalise_close_tab (*currentpage*)

finalise_restart (*currentpage*)

```

property force_full_buffered_reprogram
get_all_save_data ()
get_builtin_save_data ()
    Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.
get_channel (channel)
get_child_from_connection_table (parent_device_name, port)
get_front_panel_properties ()
get_front_panel_values ()
get_property (property)
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_builtin_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
send_clear (*args, **kwargs)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
status_monitor (*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check (support)
supports_smart_programming (support)

```

```
transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)
```

SR865.blacs_worker

```
class naqslab_devices.SR865.blacs_worker.SR865Worker(*args, **kwargs)
    Bases: naqslab_devices.VISA.blacs_worker.VISAWorker

    program_string = 'OFLT {:d};SCAL {:d};PHAS {:.6f}'
    read_string = 'OFLT?;SCAL?;PHAS?'

    phase_parser(phase_string)
        Phase Query string parser

    coerce_tau(tau_constant)
        Returns coerced, valid integer setting. Tau value rounds up. Returns max or min valid setting if out of bound.

    coerce_sens(sensitivity)
        Returns coerced, valid integer setting. Sens value rounds down. Returns max or min valid setting if out of bound.

    init()
        Initializes basic worker and opens VISA connection to device.

        Default connection timeout is 2 seconds

    check_remote_values()
        Queries the current settings for all three parameters. Parses results to actual numbers and returns.

    program_manual(front_panel_values)
        Performans manual updates from BLACS front panel. Tau and Sensitivity settings are coerced to nearest allowed value

    transition_to_buffered(device_name, h5file, initial_values, fresh)
        Stores various device handles for use in transition_to_manual method.

        Automatically called by BLACS. Should be over-ridden by inheritors.

        Parameters
            • device_name (str) – Name of device from connectiontable
            • h5file (str) – path to shot h5_file
            • initial_values (dict) – Contains the start of shot values
            • fresh (bool) – Indicates if smart_programming should be refreshed this shot

    check_status()
        Queries device state using the ESR register. Bit definitions defined in blacs_tab

    abort_buffered()
        Special abort shot code belongs here.

    abort_transition_to_buffered()
        Special abort shot configuration code belongs here.

    clear(value)
        Sends standard *CLR to clear registers of device.

        Parameters value (bool) – value of Clear button in STBstatus.ui widget

    convert_register(register)
        Converts returned register value to dict of bools
```

Parameters `register` (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_manual (*abort=False*)

Simple `transition_to_manual` method where no data is saved.

SR865.labscript_device

class `naqslab_devices.SR865.labscript_device.SR865` (*name, VISA_name*)

Bases: `naqslab_devices.VISA.labscript_device.VISA`

VISA_name can be full VISA connection string or NI-MAX alias

description = 'SR865 Lock-In Amplifier'

allowed_children = None

tau = None

sens = None

phase = None

set_tau (*tau_constant*)

Set the time constant in seconds. Uses `numpy digitize` to translate to int values. Using `digitize` corrects for round-off errors and coerces input to nearest allowed setting.

set_sens (*sensitivity*)

Set the sensitivity in Volts Uses `numpy digitize` to translate to int values. Using `digitize` corrects for round-off errors and coerces input to nearest allowed setting.

set_phase (*phase*)

Set the phase reference in degrees Device auto-converts to -180,180 range

generate_code (*hdf5_file*)

Generates the transition to buffered code in the h5 file. If parameter is not specified in shot, NaN and -1 values are set to tell worker not to change the value when programming.

add_device (*device*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into properties_dict

property_names is a dictionary {key:val,...} where each **val** is a list [var1, var2, ...] of variables to be pulled from properties_dict and added to the property with name key (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

SR865.register_classes

3.9 TektronixTDS

3.9.1 Overview

This covers the TDS series of Tektronix oscilloscopes.

naqslab_devices.TektronixTDS.

blacs_tab

naqslab_devices.TektronixTDS.

blacs_worker

naqslab_devices.TektronixTDS.

labscript_device

naqslab_devices.TektronixTDS.

register_classes

3.9.2 Detailed Documentation of naqslab_devices.TektronixTDS

TektronixTDS.blacs_tab

class naqslab_devices.TektronixTDS.blacs_tab.**TDS_ScopeTab** (**args, **kwargs*)

Bases: *naqslab_devices.VISA.blacs_tab.VISATab*

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

status_byte_labels = {'bit 0': 'Unused', 'bit 1': 'Unused', 'bit 2': 'Query Error'}

initialise_GUI ()

Loads the standard STBstatus.ui widget and sets the worker defined in __init__

ICON_BUSY = ':/qtutils/fugue/hourglass'

```

ICON_ERROR = ':/qtutils/fugue/exclamation'
ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
ICON_OK = ':/qtutils/fugue/tick'
STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
abort_buffered (*args, **kwargs)
abort_transition_to_buffered (*args, **kwargs)
add_secondary_worker (worker)
auto_create_widgets ()
auto_place_widgets (*args)
check_remote_values (*args, **kwargs)
check_time ()
clean_ui_on_restart ()
close_tab (finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver (function)
continue_restart (currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs (analog_properties)
create_analog_widgets (channel_properties)
create_dds_outputs (dds_properties)
create_dds_widgets (channel_properties)
create_device_properties (device_properties)
create_digital_outputs (digital_properties)
create_digital_widgets (channel_properties)
create_image_outputs (image_properties)
create_image_widgets (channel_properties)
create_property_widgets (device_properties)
create_worker (name, WorkerClass, workerargs=None)
    Set up a worker process. WorkerClass can be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.
property device_name
disconnect_restart_receiver (function)
property error_message
finalise_close_tab (currentpage)
finalise_restart (currentpage)

```

```
property force_full_buffered_reprogram
get_all_save_data ()
get_builtin_save_data ()
    Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.
get_channel (channel)
get_child_from_connection_table (parent_device_name, port)
get_front_panel_properties ()
get_front_panel_values ()
get_property (property)
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_built_in_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
send_clear (*args, **kwargs)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()
status_monitor (*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check (support)
supports_smart_programming (support)
```

```

transition_to_buffered(*args, **kwargs)
transition_to_manual(*args, **kwargs)
update_from_settings(settings)

```

TektronixTDS.blacs_worker

```

class naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker(*args,
                                                                **kwargs)
    Bases: naqslab_devices.VISA.blacs_worker.VISAWorker
    setup_string = ':HEADER OFF;*ESE 60;*SRE 32;*CLS;:DAT:ENC RIB;WID 2;'
    read_y_parameters_string = ':DAT:SOU CH%d;:WFMPRE:YZE?;YMU?;YOFF?'
    read_x_parameters_string = ':WFMPRE:XZE?;XIN?'
    read_waveform_string = 'CURV?'
    waveform_parser(raw_waveform_array, y0, dy, yoffset)
        Parses the numpy array from the CURV? query.
    init()
        Initializes basic worker and opens VISA connection to device.
        Default connection timeout is 2 seconds
    transition_to_manual(abort=False)
        Simple transition_to_manual method where no data is saved.
    check_status()
        Uses the more informative ESR register.
    abort_buffered()
        Special abort shot code belongs here.
    abort_transition_to_buffered()
        Special abort shot configuration code belongs here.
    check_remote_values()
    clear(value)
        Sends standard *CLR to clear registers of device.
        Parameters value (bool) – value of Clear button in STBstatus.ui widget
    convert_register(register)
        Converts returned register value to dict of bools
        Parameters register (int) – Status register value returned from read_stb
        Returns Status byte dictionary as formatted in VISATab
        Return type dict
    interrupt_startup(reason='Process.interrupt_startup() called')
        Called from the parent process. Interrupt all blocking operations on starting the child process, causing
        Process.start() to raise Interrupted(reason). After interruption, self.child may be None if startup was
        interrupted before the child was started, otherwise self.child will be the child Popen object, which
        could be at any stage of setting up its connection with the parent. This method may be called multiple
        times without raising an exception, it will simply do nothing if startup has previously been interrupted
    mainloop()
    program_manual(front_panel_values)
        Over-ride this method if remote programming is supported.
        Returns VISAWorker.check_remote_values()

```

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the run() method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring self.child exists or is None if startup was interrupted before the process was created. Then if the child is not None, call Popen.terminate() and Popen.wait() on it.

transition_to_buffered (*device_name, h5file, initial_values, fresh*)

Stores various device handles for use in transition_to_manual method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file
- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

TektronixTDS.labscript_device

```
class naqslab_devices.TektronixTDS.labscript_device.TDS_Scope (name,  
                                                             VISA_name,  
                                                             trig-  
                                                             ger_device,  
                                                             trig-  
                                                             ger_connection,  
                                                             **kwargs)
```

Bases: labscript.labscript.TriggerableDevice

VISA_name can be full VISA connection string or NI-MAX alias. Trigger Device should be fast clocked device.

description = 'Tektronics TDS Series Digital Oscilloscope'

allowed_children = [**<class 'naqslab_devices.ScopeChannel'>**]

trigger_duration = 0.001

generate_code (*hdf5_file*)

acquire (*start_time*)

Call to define time when trigger will happen for scope.

add_device (*device*)

do_checks ()

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name, location=None, *args, **kwargs*)

init_device_group (*hdf5_file*)

```

minimum_recovery_time = 0
property parent_clock_line
property pseudoclock_device
quantise_to_pseudoclock (times)
set_properties (properties_dict, property_names, overwrite=False)
    Add one or a bunch of properties packed into properties_dict
    property_names is a dictionary {key:val, ...} where each val is a list [var1, var2, ...] of variables
    to be pulled from properties_dict and added to the property with name key (it's location)
set_property (name, value, location=None, overwrite=False)
property t0
    The earliest time output can be commanded from this device at the start of the experiment. This is
    nonzero on secondary pseudoclock devices due to triggering delays.
trigger (t, duration)
    Request parent trigger device to produce a trigger at time t with given duration.
trigger_edge_type = 'rising'

```

TektronixTDS.register_classes

3.10 KeysightDCSupply

3.10.1 Overview

This device class controls Keysight/Agilent DC Power Supplies.

The underlying implementation is to use the outputs as StaticAnalogOut. The type of analog quantity (voltage or current source) is configured at the device level based on whether the outputs are voltage limited or current limited.

At present, this driver is fairly limited in its control. Most importantly, the driver does not set both voltage and current limits (i.e. setting a max current limit when in constant voltage mode). A modified StaticAnalogOut will be necessary. It is also only tested for the E364x series single output supplies. It is written such that multiple outputs could be supported easily, but that functionality is untested.

```

naqslab_devices.KeysightDCSupply.
blacs_tab
naqslab_devices.KeysightDCSupply.
blacs_worker
naqslab_devices.KeysightDCSupply.
labscript_device
naqslab_devices.KeysightDCSupply.
register_classes

```

3.10.2 Detailed Documentation of naqslab_devices.KeysightDCSupply

KeysightDCSupply.blacs_tab

```

class naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab (*args,
                                                                    **kwargs)

```

Bases: `naqslab_devices.VISA.blacs_tab.VISATab`

You MUST override this method in order to define the device worker. You then call this parent method to finish initialization.

```
status_byte_labels = {'bit 0': 'Constant Current Mode', 'bit 1': 'Constant Voltage'}
initialise_GUI()
    Loads the standard STBstatus.ui widget and sets the worker defined in __init__
ICON_BUSY = ':/qtutils/fugue/hourglass'
ICON_ERROR = ':/qtutils/fugue/exclamation'
ICON_FATAL_ERROR = ':/qtutils/fugue/exclamation-red'
ICON_OK = ':/qtutils/fugue/tick'
STBui_path = 'C:\\\\labscript_suite\\\\naqslab_devices\\\\VISA\\\\STBstatus.ui'
abort_buffered(*args, **kwargs)
abort_transition_to_buffered(*args, **kwargs)
add_secondary_worker(worker)
auto_create_widgets()
auto_place_widgets(*args)
check_remote_values(*args, **kwargs)
check_time()
clean_ui_on_restart()
close_tab(finalise=True)
    Close the tab, terminate subprocesses and join the mainloop thread. If finalise=False, then do not terminate subprocesses or join the mainloop. In this case, callers must manually call finalise_close_tab() to perform these potentially blocking operations
connect_restart_receiver(function)
continue_restart(currentpage)
    Called in a thread for the stages of restarting that may be blocking, so as to not block the main thread. Calls subsequent GUI operations in the main thread once finished blocking.
create_analog_outputs(analog_properties)
create_analog_widgets(channel_properties)
create_dds_outputs(dds_properties)
create_dds_widgets(channel_properties)
create_device_properties(device_properties)
create_digital_outputs(digital_properties)
create_digital_widgets(channel_properties)
create_image_outputs(image_properties)
create_image_widgets(channel_properties)
create_property_widgets(device_properties)
create_worker(name, WorkerClass, workerargs=None)
    Set up a worker process. WorkerClass can either be a subclass of Worker, or a string containing a fully qualified import path to a worker. The latter is useful if the worker class is in a separate file with global imports or other import-time behaviour that is undesirable to have run in the main process, for example if the imports may not be available to the main process (as may be the case once remote worker processes are implemented and the worker may be on a separate computer). The worker process will not be started immediately, it will be started once the state machine mainloop begins running. This way errors in startup will be handled using the normal state machine machinery.
property device_name
```

```

disconnect_restart_receiver (function)
property error_message
finalise_close_tab (currentpage)
finalise_restart (currentpage)
property force_full_buffered_reprogram
get_all_save_data ()
get_builtin_save_data ()
    Get builtin settings to be restored like whether the terminal is visible. Not to be overridden.
get_channel (channel)
get_child_from_connection_table (parent_device_name, port)
get_front_panel_properties ()
get_front_panel_values ()
get_property (property)
get_save_data ()
get_tab_layout ()
hide_error ()
initialise_workers ()
mainloop ()
property mode
on_force_full_buffered_reprogram ()
on_resolve_value_inconsistency ()
property primary_worker
program_device (*args, **kwargs)
program_device_properties (*args, **kwargs)
queue_work (worker_process, worker_function, *args, **kwargs)
restart (*args)
restore_builtin_save_data (data)
    Restore builtin settings to be restored like whether the terminal is visible. Not to be overridden.
restore_save_data (data)
send_clear (*args, **kwargs)
set_tab_icon_and_colour ()
    Set the tab icon and the colour of its text to the values of self._tab_icon and self._tab_text_colour
    respectively
set_terminal_visible (visible)
shutdown_workers (*args, **kwargs)
start_run (*args, **kwargs)
property state
statemachine_timeout_add (delay, statefunction, *args, **kwargs)
statemachine_timeout_remove (statefunction)
statemachine_timeout_remove_all ()

```

```

status_monitor (*args, **kwargs)
status_widget = 'STBstatus.ui'
supports_remote_value_check (support)
supports_smart_programming (support)
transition_to_buffered (*args, **kwargs)
transition_to_manual (*args, **kwargs)
update_from_settings (settings)

```

KeysightDCSupply.blacs_worker

```

class naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker (*args,
                                                                              **kwargs)

```

Bases: *naqslab_devices.VISA.blacs_worker.VISAWorker*

```

esr_mask = 60
qsr_mask = 1539
init_string = '*ESE 60;STAT:QUES:ENAB 1539;*CLS'
ident_string = 'E364'
write_both_string = 'APPL %.5f, %.5f'
write_volt_string = 'VOLT %.5f'
write_current_string = 'CURR %.5f'
read_string = 'APPL?'

```

read_parser (*response*)

Parses the Voltage & Amplitude response string

Parameters *response* (*str*) – Instrument response to current voltage/current query. Has format of “d.ddddd, d.ddddd”

Returns

containing

V (float): Current Voltage Setting A (float): Current Current Setting

Return type (tuple)

init ()

Initializes basic worker and opens VISA connection to device.

Default connection timeout is 2 seconds

check_remote_values ()

program_manual (*front_panel_values*)

Over-ride this method if remote programming is supported.

Returns *VISAWorker.check_remote_values* ()

transition_to_buffered (*device_name*, *h5file*, *initial_values*, *fresh*)

Stores various device handles for use in transition_to_manual method.

Automatically called by BLACS. Should be over-ridden by inheritors.

Parameters

- **device_name** (*str*) – Name of device from connectiontable
- **h5file** (*str*) – path to shot h5_file

- **initial_values** (*dict*) – Contains the start of shot values
- **fresh** (*bool*) – Indicates if smart_programming should be refreshed this shot

check_status ()

Customised check status for Keysight DC supplies.

This method combines flags from the event status register and the questionable status register.

merge_registers (*esr, qsr, cond*)

Merges the ESR & QSR registers with the condition into a hybrid register and converts to a dictionary for display on the BLACStab.

Parameters

- **esr** (*int*) – Value of the Event Status Register
- **qsr** (*int*) – Value of the Questionable Status Register
- **cond** (*int*) – Response of STATUS:QUESTIONABLE:CONDITION?

Returns Dictionary of values from the esr and qsr registers.

Return type return_vals (dict)

abort_buffered ()

Special abort shot code belongs here.

abort_transition_to_buffered ()

Special abort shot configuration code belongs here.

clear (*value*)

Sends standard *CLR to clear registers of device.

Parameters **value** (*bool*) – value of Clear button in STBstatus.ui widget

convert_register (*register*)

Converts returned register value to dict of bools

Parameters **register** (*int*) – Status register value returned from `read_stb`

Returns Status byte dictionary as formatted in VISATab

Return type dict

interrupt_startup (*reason='Process.interrupt_startup() called'*)

Called from the parent process. Interrupt all blocking operations on starting the child process, causing `Process.start()` to raise `Interrupted(reason)`. After interruption, `self.child` may be `None` if startup was interrupted before the child was started, otherwise `self.child` will be the child `Popen` object, which could be at any stage of setting up its connection with the parent. This method may be called multiple times without raising an exception, it will simply do nothing if startup has previously been interrupted

mainloop ()

run (*worker_name, device_name, extraargs*)

The method that gets called in the subprocess. To be overridden by subclasses

shutdown ()

Closes VISA connection to device.

start (**args, **kwargs*)

Call in the parent process to start a subprocess. Passes args and kwargs to the `run()` method

terminate (*wait_timeout=None, **kwargs*)

Interrupt process startup if not already done, ensuring `self.child` exists or is `None` if startup was interrupted before the process was created. Then if the child is not `None`, call `Popen.terminate()` and `Popen.wait()` on it.

transition_to_manual (*abort=False*)

Simple `transition_to_manual` method where no data is saved.

KeysightDCSupply.labscript_device

```
class naqslab_devices.KeysightDCSupply.labscript_device.KeysightDCSupply(name,  
                                                                           VISA_name,  
                                                                           range='LOW',  
                                                                           volt_limits=(0,  
                                                         1),  
                                                                           current_limits=(0,  
                                                         1),  
                                                                           limited='volt')
```

Bases: `naqslab_devices.VISA.labscript_device.VISA`

Keysight DC Power Supply

The labscript_device for Keysight DC Power supplies. Currently only tested for E364xA single output series devices.

Parameters

- **name** (*str*) – labscript name to assign to device. Must be an allowed python variable name.
- **VISA_name** (*str*) – VISA connection string to device. Can be alias configured in NI-MAX.
- **range** (*str*) – configures which voltage range to use. Default is 'LOW'.
- **volt_limits** (*iterable*) – voltage limits, in volts
- **current_limits** (*iterable*) – current limits, in amps
- **limited** (*str*) – Sets whether output is configured to be voltage or current limited. Default is 'volt'

description = 'DC Power Supply'

allowed_children = [`<class 'labscript.labscript.StaticAnalogOut'>`]

allowed_outputs = [0]

quantise_volt (*data*, *output*)

Quantize the currents in units of V and check it's within bounds

quantise_current (*data*, *output*)

Quantize the currents in units of A and check it's within bounds

generate_code (*hdf5_file*)

Method to generate instructions for blacs_worker to program device.

Must be over-ridden.

add_device (*device*)

get_all_children ()

get_all_outputs ()

get_properties (*location=None*)

Get all properties in location

If location is None we return all keys

get_property (*name*, *location=None*, **args*, ***kwargs*)

init_device_group (*hdf5_file*)

property parent_clock_line

property pseudoclock_device

quantise_to_pseudoclock (*times*)

set_properties (*properties_dict, property_names, overwrite=False*)

Add one or a bunch of properties packed into *properties_dict*

property_names is a dictionary {**key:val, ...**} where each **val** is a list [var1, var2, ...] of variables to be pulled from *properties_dict* and added to the property with name **key** (it's location)

set_property (*name, value, location=None, overwrite=False*)

property t0

The earliest time output can be commanded from this device at the start of the experiment. This is nonzero on secondary pseudoclock devices due to triggering delays.

KeysightDCSupply.register_classes

BUILDING DOCUMENTATION

The API documentation for this library leverages the [Sphinx](#) automatic python documentation generator. The general structure is to use the [Sphinx-apidoc](#) infrastructure to read source code docstrings to automatically build the function/class reference documentation for each device. Hand-written ReStructuredText files ([quick syntax guide](#)) are then used to provide high-level documentation that imports the auto-generated documentation. A makefile is provided to run the correct `apidoc` and `sphinx-build` commands.

4.1 Sphinx Environment

In order to build the documentation from scratch, a functioning `labscript` environment is needed with `sphinx` also installed. Because `sphinx` require a large amount of dependencies, it is recommended to copy your local `labscript` environment then install and use `sphinx` from there.

`Sphinx` can be installed into the `labscript` environment by installing the packages: `sphinx>=2.2` and `sphinx_rtd_theme`. This will allow building of the html documentation.

If you also wish to build the pdf documentation, you must also install `perl` and a latex environment (such as MikTeX for windows). The latex build is controlled using the `latexmk` latex package and requires a great many other latex packages. A partial list of required latex packages is: `cmap`, `fncychap`, `tabulary`, `parskip`, `capt-of`.

4.2 Sphinx Build

The documentation build is automated through makefiles. All commands are run from the `doc` subfolder.

The automated documentation build is performed using

```
make apidoc
```

The auto-generated files are placed in the `_apidoc` subfolder of `doc` and are given the import name of the module with the file suffix `.inc`.

The complete documentation is built using `sphinx-build` and currently supports two targets: `html` and `latex-pdf`. They are run using

```
make html
#or
make latexpdf
```

4.3 Adding Documentation

The main documentation tree is found in `index.rst`. New devices should be added using `devices.rst`. Each device should have it's own `rst` file that provides some high-level documentation and includes the auto-generated `apidoc` file. All modules and submodules in the top-level directory of the library will have documenta-

tion auto-generated using apidoc. It is up to the user to include those files where appropriate in the documentation. The include statement is of the form:

```
.. include:: _apidoc\naqslab_devices.NovaTechDDS.inc
```

INDICES AND TABLES

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

CREDITS

Authors David Meyer Zac Castillo

Licence Simplified BSD License

Version 0.2.7

PYTHON MODULE INDEX

n

[naqslab_devices](#), [7](#)
[naqslab_devices.KeysightDCSupply](#), [105](#)
[naqslab_devices.KeysightDCSupply.blacs_tab](#), [105](#)
[naqslab_devices.KeysightDCSupply.blacs_worker](#), [108](#)
[naqslab_devices.KeysightDCSupply.labscrip_device](#), [110](#)
[naqslab_devices.KeysightDCSupply.register_classes](#), [111](#)
[naqslab_devices.KeysightXSeries](#), [76](#)
[naqslab_devices.KeysightXSeries.blacs_tab](#), [77](#)
[naqslab_devices.KeysightXSeries.blacs_worker](#), [79](#)
[naqslab_devices.KeysightXSeries.labscrip_device](#), [81](#)
[naqslab_devices.KeysightXSeries.register_classes](#), [82](#)
[naqslab_devices.NovaTechDDS](#), [82](#)
[naqslab_devices.NovaTechDDS.blacs_tab](#), [82](#)
[naqslab_devices.NovaTechDDS.blacs_worker](#), [89](#)
[naqslab_devices.NovaTechDDS.labscrip_device](#), [91](#)
[naqslab_devices.NovaTechDDS.register_classes](#), [95](#)
[naqslab_devices.NovaTechDDS.runviewer_parser](#), [95](#)
[naqslab_devices.PulseBlaster_No_DDS_200](#), [71](#)
[naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab](#), [72](#)
[naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker](#), [74](#)
[naqslab_devices.PulseBlaster_No_DDS_200.labscrip_device](#), [75](#)
[naqslab_devices.PulseBlaster_No_DDS_200.register_classes](#), [76](#)
[naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser](#), [76](#)
[naqslab_devices.PulseBlasterESRPro300](#), [66](#)
[naqslab_devices.PulseBlasterESRPro300.blacs_tab](#), [66](#)
[naqslab_devices.PulseBlasterESRPro300.blacs_worker](#), [69](#)
[naqslab_devices.PulseBlasterESRPro300.labscrip_device](#), [69](#)
[naqslab_devices.PulseBlasterESRPro300.register_classes](#), [71](#)
[naqslab_devices.PulseBlasterESRPro300.runviewer_parser](#), [71](#)
[naqslab_devices.SignalGenerator.BLACS.HP_8642A](#), [16](#)
[naqslab_devices.SignalGenerator.BLACS.HP_8643A](#), [20](#)
[naqslab_devices.SignalGenerator.BLACS.HP_8648](#), [24](#)
[naqslab_devices.SignalGenerator.BLACS.KeysightSig](#), [46](#)
[naqslab_devices.SignalGenerator.BLACS.RS_SMA100B](#), [35](#)
[naqslab_devices.SignalGenerator.BLACS.RS_SMF100A](#), [39](#)
[naqslab_devices.SignalGenerator.BLACS.RS_SMHU](#), [42](#)
[naqslab_devices.SignalGenerator.blacs_tab](#), [61](#)
[naqslab_devices.SignalGenerator.blacs_worker](#), [63](#)
[naqslab_devices.SignalGenerator.labscrip_device](#), [65](#)
[naqslab_devices.SignalGenerator.Models](#), [53](#)
[naqslab_devices.SignalGenerator.register_classes](#), [66](#)
[naqslab_devices.SR865](#), [95](#)
[naqslab_devices.SR865.blacs_tab](#), [95](#)
[naqslab_devices.SR865.blacs_worker](#), [98](#)
[naqslab_devices.SR865.labscrip_device](#), [99](#)
[naqslab_devices.SR865.register_classes](#), [100](#)
[naqslab_devices.TektronixTDS](#), [100](#)
[naqslab_devices.TektronixTDS.blacs_tab](#), [100](#)
[naqslab_devices.TektronixTDS.blacs_worker](#), [103](#)
[naqslab_devices.TektronixTDS.labscrip_device](#), [103](#)

104
naqslab_devices.TektronixTDS.register_classes,
105
naqslab_devices.VISA, 10
naqslab_devices.VISA.blacs_tab, 11
naqslab_devices.VISA.blacs_worker, 13
naqslab_devices.VISA.labscrip_device,
14
naqslab_devices.VISA.register_classes,
15

INDEX

A

`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method`), 19
`method`), 106
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker`
`method`), 109
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 23
`method`), 77
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`method`), 24
`method`), 80
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method`), 26
`method`), 82
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`method`), 29
`method`), 84
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method`), 31
`method`), 87
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ActWorker`
`method`), 34
`method`), 89
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`
`method`), 49
`method`), 89
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker`
`method`), 47
`method`), 90
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 52
`method`), 72
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlasterNoDDS200Worker`
`method`), 35
`method`), 74
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 38
`method`), 66
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 39
`method`), 69
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`method`), 41
`method`), 16
`abort_buffered()` (naqs-

`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker` method), 90
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker` method), 43
`abort_buffered()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster` method), 43
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker` method), 45
`abort_buffered()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster` method), 45
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 74
`method`, 61
`abort_buffered()` (naqs-`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES` method), 64
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker` method), 64
`method`, 64
`abort_buffered()` (naqs-`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlaster` method), 69
`lab_devices.SR865.blacs_tab.SR865Tab` method), 96
`method`, 96
`abort_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` method), 17
`lab_devices.SR865.blacs_worker.SR865Worker` method), 98
`method`, 98
`abort_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker` method), 19
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 101
`method`, 101
`abort_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` method), 20
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker` method), 103
`method`, 103
`abort_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker` method), 23
`lab_devices.VISA.blacs_tab.VISATab` method), 11
`method`, 11
`abort_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 24
`lab_devices.VISA.blacs_worker.VISAWorker` method), 14
`method`, 14
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 26
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` method), 106
`method`, 106
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` method), 109
`lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker` method), 109
`method`, 109
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` method), 31
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` method), 77
`method`, 77
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker` method), 80
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker` method), 80
`method`, 80
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N` method), 49
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` method), 82
`method`, 82
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight` method), 47
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 84
`method`, 84
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight` method), 52
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 87
`method`, 87
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100` method), 85
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTab` method), 89
`method`, 89
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100` method), 38
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker` method), 89
`method`, 89
`abort_transition_to_buffered()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100` method), 39
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker` method), 39

`abort_transition_to_buffered()` (naqs- `method`), 94
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker` (naqs-
`method`), 41
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200Worker` (naqs-
`method`), 75
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker` (naqs-
`method`), 43
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300Worker` (naqs-
`method`), 70
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker` (naqslab_devices.ScopeChannel
`method`), 45
`method`, 8
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` `lab_devices.SignalGenerator.labscript_device.SignalGenerator`
`method`), 61
`method`), 65
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker` `lab_devices.SignalGenerator.Models.E8257N`
`method`), 64
`method`), 60
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab` `lab_devices.SignalGenerator.Models.HP_8642A`
`method`), 96
`method`), 56
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.SR865.blacs_worker.SR865Worker` `lab_devices.SignalGenerator.Models.HP_8643A`
`method`), 98
`method`), 55
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `lab_devices.SignalGenerator.Models.HP_8648A`
`method`), 101
`method`), 57
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker` `lab_devices.SignalGenerator.Models.HP_8648B`
`method`), 103
`method`), 58
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab` `lab_devices.SignalGenerator.Models.HP_8648C`
`method`), 11
`method`), 59
`abort_transition_to_buffered()` (naqs- `add_device()` (naqs-
`lab_devices.VISA.blacs_worker.VISAWorker` `lab_devices.SignalGenerator.Models.HP_8648D`
`method`), 14
`method`), 60
`acquire()` (naqslab_devices.CounterScopeChannel `add_device()` (naqs-
`method`), 8
`lab_devices.SignalGenerator.Models.RS_SMA100B`
`acquire()` (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope `add_device()` (naqs-
`method`), 81
`method`), 53
`acquire()` (naqslab_devices.ScopeChannel `lab_devices.SignalGenerator.Models.RS_SMF100A`
`method`), 8
`method`), 53
`acquire()` (naqslab_devices.TektronixTDS.labscript_device.TDS_Scope `add_device()` (naqs-
`method`), 104
`method`), 55
`add_device()` (naqs- `method`), 55
`lab_devices.CounterScopeChannel` `method`), `add_device()` (naqs-
9
`lab_devices.SR865.labscript_device.SR865`
`add_device()` (naqs- `method`), 99
`lab_devices.KeysightDCSupply.labscript_device.KeysightDCSupply` (naqslab_devices.StaticFreqAmp
`method`), 110
`method`), 10
`add_device()` (naqs- `add_device()` (naqs-
`lab_devices.KeysightXSeries.labscript_device.KeysightXScope` `lab_devices.TektronixTDS.labscript_device.TDS_Scope`
`method`), 81
`method`), 104
`add_device()` (naqs- `add_device()` (naqs-
`lab_devices.NovaTechDDS.labscript_device.NovaTech4090Bb` `lab_devices.VISA.labscript_device.VISA`
`method`), 93
`method`), 15
`add_device()` (naqs- `add_secondary_worker()` (naqs-
`lab_devices.NovaTechDDS.labscript_device.NovaTech4090BbAC` `lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method`), 92
`method`), 106
`add_device()` (naqs- `add_secondary_worker()` (naqs-
`lab_devices.NovaTechDDS.labscript_device.NovaTech440Ab` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 92
`method`), 106


```

        method), 77
add_secondary_worker() (naqslab_devices.VISA.blacs_tab.VISATab
        lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
        method), 11
        method), 82
add_secondary_worker() (naqslab_devices.CounterScopeChannel
        lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
        attribute), 9
        method), 84
add_secondary_worker() (naqslab_devices.KeysightDCSupply.labscript_device.KeysightDCSupply
        lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
        attribute), 110
        method), 87
add_secondary_worker() (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope
        lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
        attribute), 8
        method), 72
add_secondary_worker() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B
        lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
        attribute), 9
        method), 66
add_secondary_worker() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_A
        lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642A_Tab
        attribute), 91
        method), 17
add_secondary_worker() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A
        lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A_Tab
        attribute), 94
        method), 20
add_secondary_worker() (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.Pulse
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648A_Tab
        attribute), 75
        method), 24
add_secondary_worker() (naqslab_devices.PulseBlasterESRPro300.labscript_device.Pulse
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648B_Tab
        attribute), 70
        method), 27
add_secondary_worker() (naqslab_devices.ScopeChannel
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648C_Tab
        attribute), 8
        method), 29
add_secondary_worker() (naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
        attribute), 65
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648D_Tab
        attribute), 60
        method), 31
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.E8257N
        attribute), 60
        lab_devices.SignalGenerator.BLACS.KeysightSignalGeneratorE8257N_Tab
        attribute), 56
        method), 49
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.HP_8642A
        attribute), 56
        lab_devices.SignalGenerator.BLACS.KeysightSignalGeneratorE8257N_Tab
        attribute), 55
        method), 47
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.HP_8643A
        attribute), 57
        lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100B_Tab
        attribute), 57
        method), 35
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.HP_8648A
        attribute), 58
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A_Tab
        attribute), 58
        method), 39
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.HP_8648B
        attribute), 59
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU_Tab
        attribute), 59
        method), 43
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.HP_8648C
        attribute), 60
        lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
        attribute), 60
        method), 61
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.HP_8648D
        attribute), 54
        lab_devices.SR865.blacs_tab.SR865Tab
        attribute), 54
        method), 96
add_secondary_worker() (naqslab_devices.SignalGenerator.Models.RS_SMA100B
        attribute), 54
        lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
        attribute), 54
        method), 101
        lab_devices.SignalGenerator.Models.RS_SMF100A

```


<i>attribute</i>), 53	<i>amp_parser()</i> (naqs-
allowed_children (naqs-	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker
lab_devices.SignalGenerator.Models.RS_SMHU	method), 34
<i>attribute</i>), 55	<i>amp_parser()</i> (naqs-
allowed_children (naqs-	lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SR865.labscrip_device.SR865	method), 52
<i>attribute</i>), 99	<i>amp_parser()</i> (naqs-
allowed_children (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
lab_devices.StaticFreqAmp	<i>attribute</i>), method), 37
10	<i>amp_parser()</i> (naqs-
allowed_children (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
lab_devices.TektronixTDS.labscrip_device.TDS_Scope	method), 41
<i>attribute</i>), 104	<i>amp_parser()</i> (naqs-
allowed_children (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
lab_devices.VISA.labscrip_device.VISA	method), 45
<i>attribute</i>), 15	<i>amp_parser()</i> (naqs-
allowed_outputs (naqs-	lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
lab_devices.KeysightDCSupply.labscrip_device.KeysightDCSupply	method), 64
<i>attribute</i>), 110	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
lab_devices.SignalGenerator.labscrip_device.SignalGenerator	<i>attribute</i>), 19
<i>attribute</i>), 65	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
lab_devices.SignalGenerator.Models.E8257N	<i>attribute</i>), 23
<i>attribute</i>), 60	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker
lab_devices.SignalGenerator.Models.HP_8642A	<i>attribute</i>), 34
<i>attribute</i>), 56	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SignalGenerator.Models.HP_8643A	<i>attribute</i>), 51
<i>attribute</i>), 55	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
lab_devices.SignalGenerator.Models.HP_8648A	<i>attribute</i>), 37
<i>attribute</i>), 57	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
lab_devices.SignalGenerator.Models.HP_8648B	<i>attribute</i>), 41
<i>attribute</i>), 58	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
lab_devices.SignalGenerator.Models.HP_8648C	<i>attribute</i>), 45
<i>attribute</i>), 59	<i>amp_query_string</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
lab_devices.SignalGenerator.Models.HP_8648D	<i>attribute</i>), 64
<i>attribute</i>), 59	<i>amp_scale_factor</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
lab_devices.SignalGenerator.Models.RS_SMA100B	<i>attribute</i>), 19
<i>attribute</i>), 54	<i>amp_scale_factor</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
lab_devices.SignalGenerator.Models.RS_SMF100A	<i>attribute</i>), 22
<i>attribute</i>), 53	<i>amp_scale_factor</i> (naqs-
amp_limits (naqs-	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker
lab_devices.SignalGenerator.Models.RS_SMHU	<i>attribute</i>), 33
<i>attribute</i>), 54	<i>amp_scale_factor</i> (naqs-
amp_parser() (naqs-	lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker	method), 19
method), 19	<i>amp_scale_factor</i> (naqs-
amp_parser() (naqs-	lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker	method), 37
method), 23	<i>amp_scale_factor</i> (naqs-

`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`attribute`), 41 `amp_write_string` (naqs-
`amp_scale_factor` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`), 45
`attribute`), 45 `amp_write_string` (naqs-
`amp_scale_factor` (naqs-`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`), 64
`attribute`), 63 `analog_waveform_parser()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeWorker`
`lab_devices.SignalGenerator.labscript_device.SignalGeneratorWorker`), 79
`attribute`), 65 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`lab_devices.SignalGenerator.Models.E8257N` method), 106
`attribute`), 60 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`lab_devices.SignalGenerator.Models.HP_8642A` method), 77
`attribute`), 56 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.SignalGenerator.Models.HP_8643A` method), 82
`attribute`), 55 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.SignalGenerator.Models.HP_8648A` method), 84
`attribute`), 57 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.Models.HP_8648B` method), 87
`attribute`), 58 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterES`
`lab_devices.SignalGenerator.Models.HP_8648C` method), 72
`attribute`), 59 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES`
`lab_devices.SignalGenerator.Models.HP_8648D` method), 66
`attribute`), 59 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.SignalGenerator.Models.RS_SMA100B` method), 17
`attribute`), 54 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.Models.RS_SMF100A` method), 20
`attribute`), 53 `auto_create_widgets()` (naqs-
`amp_scale_factor` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.Models.RS_SMHU` method), 24
`attribute`), 54 `auto_create_widgets()` (naqs-
`amp_write_string` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`), 27
`attribute`), 19 `auto_create_widgets()` (naqs-
`amp_write_string` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`), 29
`attribute`), 23 `auto_create_widgets()` (naqs-
`amp_write_string` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`), 31
`attribute`), 34 `auto_create_widgets()` (naqs-
`amp_write_string` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker`
`attribute`), 51 `auto_create_widgets()` (naqs-
`amp_write_string` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker`
`attribute`), 37 `auto_create_widgets()` (naqs-
`amp_write_string` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`

auto_create_widgets() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
 method), 39 auto_place_widgets() (naqs-
 auto_create_widgets() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 47
 method), 43 auto_place_widgets() (naqs-
 auto_create_widgets() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 35
 method), 61 auto_place_widgets() (naqs-
 auto_create_widgets() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
 lab_devices.SR865.blacs_tab.SR865Tab method), 39
 method), 96 auto_place_widgets() (naqs-
 auto_create_widgets() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 43
 method), 101 auto_place_widgets() (naqs-
 auto_create_widgets() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
 lab_devices.VISA.blacs_tab.VISATab method), 61
 method), 11 auto_place_widgets() (naqs-
 auto_place_widgets() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
 lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 96
 method), 106 auto_place_widgets() (naqs-
 auto_place_widgets() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 101
 method), 77 auto_place_widgets() (naqs-
 auto_place_widgets() (naqs- lab_devices.VISA.blacs_tab.VISATab
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 11
 method), 82
 auto_place_widgets() (naqs- **B**
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
 method), 85 base_decimals (naqs-
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
 attribute), 16
 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
 method), 87 base_decimals (naqs-
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
 attribute), 20
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab (naqs-
 method), 72 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
 auto_place_widgets() (naqs- attribute), 24
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab (naqs-
 method), 66 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB
 auto_place_widgets() (naqs- attribute), 27
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab (naqs-
 method), 17 base_decimals
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 attribute), 29 (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
 method), 20 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB
 auto_place_widgets() (naqs- attribute), 31 (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
 method), 24 base_decimals
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
 attribute), 49 (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB
 method), 27 base_decimals
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
 attribute), 47 (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 method), 29 base_decimals
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
 attribute), 35 (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB
 method), 31 base_decimals
 auto_place_widgets() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
 attribute), 39

```

base_decimals (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
attribute), 24
base_decimals (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
attribute), 27
base_decimals (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 29
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
attribute), 16
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
attribute), 31
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
attribute), 20
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
attribute), 49
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
attribute), 47
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
attribute), 35
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab
attribute), 39
base_max (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648FTab
attribute), 43
base_max (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab
attribute), 61
base_max (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
attribute), 24
base_max (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
attribute), 35
base_max (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
attribute), 20
base_max (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
attribute), 39
base_max (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 24
base_max (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 61
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
attribute), 16
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
attribute), 27
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
attribute), 20
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
attribute), 49
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
attribute), 47
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
attribute), 35
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab
attribute), 39
base_min (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648FTab
attribute), 43
base_min (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab
attribute), 61
base_min (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
attribute), 24
base_min (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
attribute), 35
base_min (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
attribute), 20
base_min (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
attribute), 39
base_min (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 24
base_min (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 61
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
attribute), 16
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
attribute), 31
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
attribute), 20
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
attribute), 49
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
attribute), 47
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
attribute), 35
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab
attribute), 39
base_step (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648FTab
attribute), 43
base_step (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab
attribute), 61
base_step (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
attribute), 24
base_step (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
attribute), 35
base_step (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
attribute), 20
base_step (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
attribute), 39
base_step (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 24
base_step (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
attribute), 61

```


C

check_connection() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker method), 19
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker method), 89
 check_connection() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 20
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker method), 89
 check_connection() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker method), 23
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker method), 90
 check_error() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 24
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker method), 89
 check_error() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 27
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker method), 90
 check_error() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 29
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker method), 90
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 31
 check_remote_values() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 106
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 34
 check_remote_values() (naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker method), 108
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N method), 49
 check_remote_values() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 77
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight method), 47
 check_remote_values() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker method), 80
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight method), 52
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 82
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100 method), 35
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 85
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100 method), 38
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 87
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100 method), 39
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker method), 89
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100 method), 41
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker method), 90
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 43
 check_remote_values() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker method), 90
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 45
 check_remote_values() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 72
 check_remote_values() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 63
 check_remote_values() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 66
 check_remote_values() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 64
 check_remote_values() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 17

`lab_devices.SR865.blacs_tab.SR865Tab`
`method`), 96
`check_remote_values()` (`naqs-`
`lab_devices.SR865.blacs_worker.SR865Worker`
`method`), 98
`check_remote_values()` (`naqs-`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method`), 101
`check_remote_values()` (`naqs-`
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`method`), 103
`check_remote_values()` (`naqs-`
`lab_devices.VISA.blacs_tab.VISATab`
`method`), 11
`check_remote_values()` (`naqs-`
`lab_devices.VISA.blacs_worker.VISAWorker`
`method`), 13
`check_status()` (`naqs-`
`lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker`
`method`), 109
`check_status()` (`naqs-`
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`method`), 80
`check_status()` (`naqs-`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker`
`method`), 74
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 69
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`
`method`), 19
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`
`method`), 23
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`
`method`), 34
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker`
`method`), 52
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker`
`method`), 37
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`method`), 41
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGen`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`
`method`), 45
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker`
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`
`method`), 64
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`lab_devices.SR865.blacs_worker.SR865Worker`
`method`), 98
`check_status()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`method`), 43
`method`), 103
`check_status()` (`naqs-`
`lab_devices.VISA.blacs_worker.VISAWorker`
`method`), 13
`check_time()` (`naqs-`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method`), 106
`check_time()` (`naqs-`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 77
`check_time()` (`naqs-`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method`), 83
`check_time()` (`naqs-`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`method`), 85
`check_time()` (`naqs-`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method`), 85
`check_time()` (`naqs-`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method`), 85
`check_time()` (`naqs-`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS200Worker`
`method`), 72
`check_time()` (`naqs-`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Worker`
`method`), 72
`check_time()` (`naqs-`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Worker`
`method`), 72
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 69
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`
`method`), 20
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`
`method`), 24
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`
`method`), 27
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker`
`method`), 52
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker`
`method`), 37
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`method`), 41
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGen`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`
`method`), 47
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker`
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`
`method`), 35
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`lab_devices.SR865.blacs_worker.SR865Worker`
`method`), 39
`check_time()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`method`), 43

`check_time()` (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab method), 35
`check_time()` (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 62
`check_time()` (naqslab_devices.SR865.blacs_tab.SR865Tab method), 39
`check_time()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 96
`check_time()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 43
`check_time()` (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 101
`check_time()` (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 62
`check_time()` (naqslab_devices.VISA.blacs_tab.VISATab method), 11
`clean_ui_on_restart()` (naqslab_devices.SR865.blacs_tab.SR865Tab method), 96
`clean_ui_on_restart()` (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 106
`clean_ui_on_restart()` (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 101
`clean_ui_on_restart()` (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 77
`clean_ui_on_restart()` (naqslab_devices.VISA.blacs_tab.VISATab method), 11
`clean_ui_on_restart()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 83
`clean_ui_on_restart()` (naqslab_devices.NovaTech409B.Tab method), 109
`clean_ui_on_restart()` (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeTab method), 80
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 19
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 23
`clean_ui_on_restart()` (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200.Tab method), 34
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 28
`clean_ui_on_restart()` (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300.Tab method), 67
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab method), 38
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 42
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 45
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 45
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorTab method), 64
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 98
`clean_ui_on_restart()` (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorkerTab method), 103
`clean_ui_on_restart()` (naqslab_devices.VISA.blacs_worker.VISAWorkerTab method), 14
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 29
`clean_ui_on_restart()` (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B.Tab method), 93
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 31
`clean_ui_on_restart()` (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_ACTab method), 91
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N.Tab method), 49
`clean_ui_on_restart()` (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A.Tab method), 94
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab method), 47
`clean_ui_on_restart()` (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B.Tab method), 93

clock_limit (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
 lab_devices.NovaTechDDS.labscript_device.NovaTech409B_ACTab method), 49
 attribute), 91 close_tab() (naqs-
 clock_limit (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
 lab_devices.NovaTechDDS.labscript_device.NovaTech440ATab method), 47
 attribute), 94 close_tab() (naqs-
 clock_limit (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
 lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
 attribute), 75 close_tab() (naqs-
 clock_limit (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
 lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
 attribute), 70 close_tab() (naqs-
 clock_resolution (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
 lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
 attribute), 75 close_tab() (naqs-
 clock_resolution (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
 lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
 attribute), 70 close_tab() (naqs-
 close_tab() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
 lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 96
 method), 106 close_tab() (naqs-
 close_tab() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 101
 method), 77 close_tab() (naqs-
 close_tab() (naqs- lab_devices.VISA.blacs_tab.VISATab
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 11
 method), 83 coerce_sens() (naqs-
 close_tab() (naqs- lab_devices.SR865.blacs_worker.SR865Worker
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 98
 method), 85 coerce_tau() (naqs-
 close_tab() (naqs- lab_devices.SR865.blacs_worker.SR865Worker
 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 98
 method), 87 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
 method), 72 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
 method), 67 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 83
 method), 17 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 85
 method), 21 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 87
 method), 24 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 72
 method), 27 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 67
 method), 29 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 17
 method), 32 connect_restart_receiver() (naqs-
 close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab

method), 21
 connect_restart_receiver() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 67
 method), 24
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648A.HP_8648ATab method), 17
 method), 27
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 17
 method), 27
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 21
 method), 29
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 24
 method), 32
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 27
 method), 49
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 47
 method), 47
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 35
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N.HP_8648CTab method), 39
 method), 39
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab method), 47
 method), 43
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab method), 35
 method), 62
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 39
 method), 96
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 43
 method), 101
 connect_restart_receiver() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 62
 method), 11
 continue_restart() (naqs- lab_devices.SR865.blacs_tab.SR865Tab method), 96
 method), 106
 continue_restart() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 101
 method), 77
 continue_restart() (naqs- lab_devices.VISA.blacs_tab.VISATab method), 11
 method), 83
 continue_restart() (naqs- lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyTab method), 109
 method), 85
 continue_restart() (naqs- lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorkerTab method), 80
 method), 87
 continue_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor method), 72
 method), 72

`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`
`method`), 23

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`
`method`), 34

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker`
`method`), 52

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker`
`method`), 38

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`method`), 42

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`
`method`), 46

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorTab`
`method`), 64

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SR865.blacs_worker.SR865Worker`
`method`), 98

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`method`), 103

`convert_register()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.VISA.blacs_worker.VISAWorker`
`method`), 13

`convert_to_pb_inst()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200`
`method`), 75

`convert_to_pb_inst()` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300`
`method`), 70

`core_clock_freq` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker`
`attribute`), 74

`core_clock_freq` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200`
`attribute`), 75

`core_clock_freq` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`attribute`), 69

`core_clock_freq` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300`
`attribute`), 70

`count()` (`naqslab_devices.CounterScopeChannel` `create_analog_outputs()` (`naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method`), 8

`CounterScopeChannel` (`class` `in` `naqslab_devices`), 8

`create_analog_outputs()` (`naqslab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method`), 106

`create_analog_outputs()` (`naqslab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method`), 77

`create_analog_outputs()` (`naqslab_devices.VISA.blacs_tab.VISATab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 101

```

method), 11
create_analog_widgets() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab), 96
method), 106
create_analog_widgets() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab), 101
method), 77
create_analog_widgets() (naqs- lab_devices.VISA.blacs_tab.VISATab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab), 11
method), 83
create_analog_widgets() (naqs- lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab), 106
method), 85
create_analog_widgets() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab), 77
method), 87
create_analog_widgets() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab), 83
method), 72
create_analog_widgets() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab), 83
method), 67
create_analog_widgets() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab), 87
method), 17
create_analog_widgets() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab), 72
method), 21
create_analog_widgets() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 67
method), 25
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 17
method), 27
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 21
method), 29
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 25
method), 32
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 27
method), 49
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab), 27
method), 47
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab), 36
method), 36
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab), 39
method), 39
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 47
method), 43
create_analog_widgets() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab), 36
method), 62
create_dds_outputs() (naqs-

```

```

        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
        method), 39
create_dds_outputs() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 47
        method), 43
create_dds_outputs() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
        lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 36
        method), 62
create_dds_outputs() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
        lab_devices.SR865.blacs_tab.SR865Tab method), 39
        method), 96
create_dds_outputs() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
        lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 43
        method), 101
create_dds_outputs() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
        lab_devices.VISA.blacs_tab.VISATab method), 62
        method), 12
create_dds_widgets() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
        lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 96
        method), 106
create_dds_widgets() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
        lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 101
        method), 77
create_dds_widgets() (naqs- lab_devices.VISA.blacs_tab.VISATab
        lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 12
        method), 83
create_dds_widgets() (naqs- lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
        lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 106
        method), 85
create_dds_widgets() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
        lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 78
        method), 87
create_dds_widgets() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
        lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 72
        method), 72
create_dds_widgets() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
        lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 67
        method), 67
create_dds_widgets() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
        lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 87
        method), 17
create_dds_widgets() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
        lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 72
        method), 21
create_dds_widgets() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300_Tab
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 67
        method), 25
create_dds_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 17
        method), 27
create_dds_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 21
        method), 29
create_dds_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 25
        method), 32
create_dds_widgets() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab method), 27

```


`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`method`), 21
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`method`), 29
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB`
`method`), 32
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`
`method`), 50
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGonTab`
`method`), 47
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`method`), 36
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB`
`method`), 36
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`method`), 39
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method`), 39
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`
`method`), 47
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method`), 43
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`method`), 62
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method`), 62
`create_device_properties()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`method`), 96
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method`), 39
`create_device_properties()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method`), 43
`create_device_properties()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method`), 101
`create_device_properties()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab`
`method`), 12
`create_digital_outputs()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`method`), 96
`create_digital_outputs()` (naqs-
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method`), 106
`create_digital_outputs()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method`), 101
`create_digital_outputs()` (naqs-
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 78
`create_digital_outputs()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab`
`method`), 12
`create_digital_outputs()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method`), 83
`create_digital_outputs()` (naqs-
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method`), 106
`create_digital_outputs()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`method`), 85
`create_digital_outputs()` (naqs-
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 78
`create_digital_outputs()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method`), 87
`create_digital_outputs()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method`), 83
`create_digital_outputs()` (naqs-
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 72
`create_digital_outputs()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`method`), 85
`create_digital_outputs()` (naqs-
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 67
`create_digital_outputs()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method`), 87
`create_digital_outputs()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`method`), 17
`create_digital_outputs()` (naqs-
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster`

`method)`, 72
`create_digital_widgets()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method)`, 67
`create_digital_widgets()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`method)`, 87
`method)`, 17
`create_digital_widgets()` (naqs-
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`method)`, 72
`method)`, 21
`create_digital_widgets()` (naqs-
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`method)`, 67
`method)`, 25
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`method)`, 17
`method)`, 27
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`method)`, 21
`method)`, 29
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`method)`, 25
`method)`, 32
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`
`method)`, 27
`method)`, 50
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab`
`method)`, 47
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`method)`, 36
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method)`, 40
`method)`, 40
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method)`, 47
`method)`, 43
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method)`, 36
`method)`, 62
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`lab_devices.SR865.blacs_tab.SR865Tab`
`method)`, 40
`method)`, 96
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method)`, 43
`method)`, 101
`create_digital_widgets()` (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.VISA.blacs_tab.VISATab`
`method)`, 62
`method)`, 12
`create_image_outputs()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method)`, 96
`method)`, 106
`create_image_outputs()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method)`, 101
`method)`, 78
`create_image_outputs()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method)`, 12
`method)`, 83
`create_image_widgets()` (naqs-

`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` method), 96
`create_image_widgets()` (naqs-`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 106
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` method), 101
`create_image_widgets()` (naqs-`lab_devices.VISA.blacs_tab.VISATab` method), 78
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` method), 12
`create_image_widgets()` (naqs-`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` method), 83
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 106
`create_image_widgets()` (naqs-`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` method), 85
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 78
`create_image_widgets()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` method), 87
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` method), 72
`create_image_widgets()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 67
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` method), 67
`create_image_widgets()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 87
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` method), 17
`create_image_widgets()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` method), 21
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` method), 72
`create_image_widgets()` (naqs-`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` method), 67
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 25
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` method), 17
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 27
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` method), 21
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` method), 29
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` method), 25
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab` method), 32
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648FTab` method), 27
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab` method), 50
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` method), 47
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab` method), 47
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` method), 36
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab` method), 36
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab` method), 40
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` method), 40
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab` method), 47
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` method), 43
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab` method), 36
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 62
`create_image_widgets()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` method), 40
`lab_devices.SR865.blacs_tab.SR865Tab` method), 40

create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab method), 47
 create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU_Tab method), 43
 create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100B_Tab method), 36
 create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A_Tab method), 40
 create_property_widgets() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 96
 create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU_Tab method), 43
 create_property_widgets() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 101
 create_property_widgets() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 62
 create_property_widgets() (naqslab_devices.VISA.blacs_tab.VISATab method), 12
 create_worker() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 96
 create_worker() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupply_Tab method), 106
 create_worker() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 101
 create_worker() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 78
 create_worker() (naqslab_devices.VISA.blacs_tab.VISATab method), 12
 create_worker() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACCTab method), 83
 create_worker() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 85
 create_worker() (naqslab_devices.CounterScopeChannel attribute), 8
 create_worker() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 87
 create_worker() (naqslab_devices.KeysightDCSupply.labscrip_device.KeysightDCSupply attribute), 110
 create_worker() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 72
 create_worker() (naqslab_devices.KeysightXSeries.labscrip_device.KeysightXScope attribute), 81
 create_worker() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 67
 create_worker() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech409B attribute), 92
 create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 17
 create_worker() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech409B_ACCTab method), 91
 create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 21
 create_worker() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech440A attribute), 94
 create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 25
 create_worker() (naqslab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200 attribute), 75
 create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 27
 create_worker() (naqslab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300 attribute), 69
 create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 30
 create_worker() (naqslab_devices.ScopeChannel attribute), 8
 create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 32
 create_worker() (naqslab_devices.SignalGenerator.labscrip_device.SignalGenerator attribute), 65
 create_worker() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab method), 50
 create_worker() (naqslab_devices.SignalGenerator.Models.E8257N attribute), 60
 create_worker() (naqslab_devices description), (naqslab_devices description)

`lab_devices.SignalGenerator.Models.HP_8642A.device_name()` (naqs-
`attribute`), 56 `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`description` (naqs-`property`), 17
`lab_devices.SignalGenerator.Models.HP_8643A.device_name()` (naqs-
`attribute`), 55 `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`description` (naqs-`property`), 21
`lab_devices.SignalGenerator.Models.HP_8648A.device_name()` (naqs-
`attribute`), 57 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`description` (naqs-`property`), 25
`lab_devices.SignalGenerator.Models.HP_8648B.device_name()` (naqs-
`attribute`), 58 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`description` (naqs-`property`), 27
`lab_devices.SignalGenerator.Models.HP_8648C.device_name()` (naqs-
`attribute`), 59 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`description` (naqs-`property`), 30
`lab_devices.SignalGenerator.Models.HP_8648D.device_name()` (naqs-
`attribute`), 59 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`description` (naqs-`property`), 32
`lab_devices.SignalGenerator.Models.RS_SMA100B.device_name()` (naqs-
`attribute`), 54 `lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`description` (naqs-`property`), 50
`lab_devices.SignalGenerator.Models.RS_SMF100A.device_name()` (naqs-
`attribute`), 53 `lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`description` (naqs-`property`), 47
`lab_devices.SignalGenerator.Models.RS_SMHU.device_name()` (naqs-
`attribute`), 54 `lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10`
`description` (naqs-`property`), 36
`lab_devices.SR865.labscript_device.SR865.device_name()` (naqs-
`attribute`), 99 `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10`
`description` (naqslab_devices.StaticFreqAmp `attribute`), 10 `property`), 40
`description` (naqs-`property`), 40
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`description` (naqs-`property`), 44
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`description` (naqs-`property`), 62
`lab_devices.VISA.labscript_device.VISA.device_name()` (naqs-
`attribute`), 15 `lab_devices.SR865.blacs_tab.SR865Tab`
`device_name()` (naqs-`property`), 96
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`device_name()` (naqs-`property`), 106
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`device_name()` (naqs-`property`), 101
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`device_name()` (naqs-`property`), 78
`lab_devices.VISA.blacs_tab.VISATab`
`device_name()` (naqs-`property`), 12
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`device_name()` (naqs-`property`), 83
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWor`
`device_name()` (naqs-`property`), 79
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`device_name()` (naqs-`property`), 85
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWor`
`device_name()` (naqs-`property`), 79
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`device_name()` (naqs-`property`), 87
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.Pulse`
`device_name()` (naqs-`property`), 72
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterNoDDS_200_Tab`
`device_name()` (naqs-`property`), 72
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBl`
`device_name()` (naqs-`property`), 300
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`property`), 67
`disable()` (naqslab_devices.StaticFreqAmp

method), 10 disconnect_restart_receiver() (naqs-
lab_devices.SR865.blacs_tab.SR865Tab
method), 96
method), 106 disconnect_restart_receiver() (naqs-
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 101
method), 78 disconnect_restart_receiver() (naqs-
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
method), 12
method), 83 disconnect_restart_receiver() (naqs-
lab_devices.VISA.blacs_tab.VISATab
method), 12
method), 83 do_checks() (naqs-
lab_devices.KeysightXSeries.labscrip_device.KeysightXScope
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 81
method), 85 do_checks() (naqs-
lab_devices.PulseBlaster_No_DDS_200.labscrip_device.Pulse
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 75
method), 87 do_checks() (naqs-
lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBl
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 73 do_checks() (naqs-
lab_devices.TektronixTDS.labscrip_device.TDS_Scope
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 67

disconnect_restart_receiver() (naqs- E
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab (class in naqs-
method), 17 lab_devices.SignalGenerator.Models),
disconnect_restart_receiver() (naqs- 60
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab (class in naqs-
method), 21 lab_devices.SignalGenerator.BLACS.KeysightSigGens),
disconnect_restart_receiver() (naqs- 49
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab (naqslab_devices.StaticFreqAmp method),
method), 25 10
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTa
method), 27 property), 107
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
method), 30 property), 78
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
method), 32 property), 83
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
method), 50 property), 85
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
method), 48 property), 87
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
method), 36 property), 73
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
method), 40 property), 67
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUATab lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
method), 44 property), 17
disconnect_restart_receiver() (naqs- error_message() (naqs-
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
method), 62 property), 21

```

error_message() (naqs- method), 83
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab.close_tab() (naqs-
property), 25 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
error_message() (naqs- method), 85
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab.close_tab() (naqs-
property), 27 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
error_message() (naqs- method), 87
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab.close_tab() (naqs-
property), 30 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
error_message() (naqs- method), 73
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab.close_tab() (naqs-
property), 32 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
error_message() (naqs- method), 67
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab.close_tab() (naqs-
property), 50 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
error_message() (naqs- method), 17
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab() (naqs-
property), 48 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
error_message() (naqs- method), 21
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab() (naqs-
property), 36 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
error_message() (naqs- method), 25
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab() (naqs-
property), 40 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
error_message() (naqs- method), 27
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab.close_tab() (naqs-
property), 44 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
error_message() (naqs- method), 30
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab.close_tab() (naqs-
property), 62 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
error_message() (naqs- method), 32
lab_devices.SR865.blacs_tab.SR865Tab finalise_close_tab() (naqs-
property), 96 lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
error_message() (naqs- method), 50
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab finalise_close_tab() (naqs-
property), 101 lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
error_message() (naqs- method), 48
lab_devices.VISA.blacs_tab.VISATab prop- finalise_close_tab() (naqs-
erty), 12 lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
error_parser() (naqs- method), 36
lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker.close_tab() (naqs-
method), 79 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
esr_mask (naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker
attribute), 108 finalise_close_tab() (naqs-
esr_mask (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
attribute), 79 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 44
ESRPro (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker (naqs-
attribute), 69 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
method), 62
F finalise_close_tab() (naqs-
finalise_close_tab() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
method), 96
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab() (naqs-
method), 107 finalise_close_tab()
finalise_close_tab() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 101
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab() (naqs-
method), 78 finalise_close_tab()
finalise_close_tab() (naqs- lab_devices.VISA.blacs_tab.VISATab
method), 12
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B ACTab()

```

```

finalise_restart() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab), 96
method), 107 finalise_restart() (naqs-
finalise_restart() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab), 101
method), 78 finalise_restart() (naqs-
finalise_restart() (naqs- lab_devices.VISA.blacs_tab.VISATab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab), 12
method), 83 flag_is_clock() (naqs-
finalise_restart() (naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.Pulse
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 75
method), 85 flag_is_clock() (naqs-
finalise_restart() (naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBl
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 70
method), 87 flag_valid() (naqs-
finalise_restart() (naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.Pulse
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 73 flag_valid() (naqs-
finalise_restart() (naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBl
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 67 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab), 107
method), 17 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab), 78
method), 21 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 83
method), 25 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 85
method), 27 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 87
method), 30 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 73
method), 32 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257PTab), 67
method), 50 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
method), 48 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab), 31
method), 36 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab), 35
method), 40 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 27
method), 44 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab), 30
method), 62 force_full_buffered_reprogram() (naqs-
finalise_restart() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab

```


property), 32

freq_limits (naqs-
lab_devices.SignalGenerator.Models.RS_SMHU
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab), 54

property), 50

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab

property), 48

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab

property), 36

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worke
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab

property), 40

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 51

property), 44

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTabmethod), 37

property), 62

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10
lab_devices.SR865.blacs_tab.SR865Tab

method), 41

property), 96

freq_parser() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWor
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab

method), 45

property), 101

freq_parser() (naqs-
lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWo
lab_devices.VISA.blacs_tab.VISATab

prop-
erty), 12

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor
lab_devices.SignalGenerator.labscrip_device.SignalGenattribute), 19

attribute), 65

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor
lab_devices.SignalGenerator.Models.E8257N

attribute), 23

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worke
lab_devices.SignalGenerator.Models.HP_8642A

attribute), 34

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SignalGenerator.Models.HP_8643A

attribute), 51

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10
lab_devices.SignalGenerator.Models.HP_8648A

attribute), 37

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10
lab_devices.SignalGenerator.Models.HP_8648B

attribute), 41

freq_query_string (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWor
lab_devices.SignalGenerator.Models.HP_8648C

attribute), 45

freq_query_string (naqs-
lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWo
lab_devices.SignalGenerator.Models.HP_8648D

attribute), 63

freq_write_string (naqs-
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor
lab_devices.SignalGenerator.Models.RS_SMA100B

attribute), 19

freq_write_string (naqs-
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor
lab_devices.SignalGenerator.Models.RS_SMF100A

attribute), 23

freq_write_string (naqs-
lab_devices.SignalGenerator.Models.RS_SMF100A

attribute), 53

`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648A` `lab_devices.SignalGenerator.Models.HP_8648A`
`attribute), 34` `method), 57`
`freq_write_string` `(naqs- generate_code ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenModels` `lab_devices.SignalGenerator.Models.HP_8648B`
`attribute), 51` `method), 58`
`freq_write_string` `(naqs- generate_code ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100B` `lab_devices.SignalGenerator.Models.HP_8648C`
`attribute), 37` `method), 59`
`freq_write_string` `(naqs- generate_code ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A` `lab_devices.SignalGenerator.Models.HP_8648D`
`attribute), 41` `method), 60`
`freq_write_string` `(naqs- generate_code ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU` `lab_devices.SignalGenerator.Models.RS_SMA100B`
`attribute), 45` `method), 54`
`freq_write_string` `(naqs- generate_code ()` `(naqs-`
`lab_devices.SignalGenerator.blacs_worker.SignalGenerator` `lab_devices.SignalGenerator.Models.RS_SMF100A`
`attribute), 63` `method), 53`
G `generate_code ()` `(naqs-`
`lab_devices.SignalGenerator.Models.RS_SMHU`
`generate_code ()` `(naqs-` `method), 55`
`lab_devices.CounterScopeChannel` `method), generate_code ()` `(naqs-`
`9` `lab_devices.SR865.labscrip_device.SR865`
`generate_code ()` `(naqs-` `method), 99`
`lab_devices.KeysightDCSupply.labscrip_device.KeysightDCSupply` `generate_code ()` `(naqs-`
`method), 110` `lab_devices.StaticFreqAmp` `method), 10`
`generate_code ()` `(naqs- generate_code ()` `(naqs-`
`lab_devices.KeysightXSeries.labscrip_device.KeysightXScope` `lab_devices.TektronixTDS.labscrip_device.TDS_Scope`
`method), 81` `method), 104`
`generate_code ()` `(naqs- generate_code ()` `(naqs-`
`lab_devices.NovaTechDDS.labscrip_device.NovaTech409B` `lab_devices.VISA.labscrip_device.VISA`
`method), 93` `method), 15`
`generate_code ()` `(naqs- generate_registers ()` `(naqs-`
`lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_AC` `lab_devices.PulseBlaster_No_DDS_200.labscrip_device.Pulse`
`method), 92` `method), 75`
`generate_code ()` `(naqs- generate_registers ()` `(naqs-`
`lab_devices.NovaTechDDS.labscrip_device.NovaTech440A` `lab_devices.PulseBlasterESRPro300.labscrip_device.Pulse`
`method), 94` `method), 70`
`generate_code ()` `(naqs- get_all_children ()` `(naqs-`
`lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200` `lab_devices.CounterScopeChannel` `method),`
`method), 75` `9`
`generate_code ()` `(naqs- get_all_children ()` `(naqs-`
`lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300` `lab_devices.KeysightDCSupply.labscrip_device.KeysightDCSu`
`method), 70` `method), 110`
`generate_code ()` `(naqslab_devices.ScopeChannel` `get_all_children ()` `(naqs-`
`method), 8` `lab_devices.KeysightXSeries.labscrip_device.KeysightXScope`
`generate_code ()` `(naqs-` `method), 81`
`lab_devices.SignalGenerator.labscrip_device.SignalGenerator` `get_all_children ()` `(naqs-`
`method), 65` `lab_devices.NovaTechDDS.labscrip_device.NovaTech409B`
`generate_code ()` `(naqs-` `method), 93`
`lab_devices.SignalGenerator.Models.E8257N` `get_all_children ()` `(naqs-`
`method), 60` `lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_AC`
`generate_code ()` `(naqs-` `method), 92`
`lab_devices.SignalGenerator.Models.HP_8642A` `get_all_children ()` `(naqs-`
`method), 56` `lab_devices.NovaTechDDS.labscrip_device.NovaTech440A`
`generate_code ()` `(naqs-` `method), 94`
`lab_devices.SignalGenerator.Models.HP_8643A` `get_all_children ()` `(naqs-`
`method), 55` `lab_devices.PulseBlaster_No_DDS_200.labscrip_device.Pulse`
`generate_code ()` `(naqs-` `method), 75`

get_all_children()	(naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300.get_all_children() method), 70	get_all_outputs()	(naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300.get_all_outputs() method), 93
get_all_children()	(naqs- lab_devices.ScopeChannel method), 8	get_all_outputs()	(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC.get_all_outputs() method), 92
get_all_children()	(naqs- lab_devices.SignalGenerator.labscript_device.SignalGenerator.get_all_children() method), 65	get_all_outputs()	(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech440A.get_all_outputs() method), 94
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.E8257N.get_all_children() method), 60	get_all_outputs()	(naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200.get_all_outputs() method), 75
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.HP_8642A.get_all_children() method), 56	get_all_outputs()	(naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300.get_all_outputs() method), 70
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.HP_8643A.get_all_children() method), 55	get_all_outputs()	(naqs- lab_devices.ScopeChannel method), 8
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.HP_8648A.get_all_children() method), 57	get_all_outputs()	(naqs- lab_devices.SignalGenerator.labscript_device.SignalGenerator.get_all_outputs() method), 65
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.HP_8648B.get_all_children() method), 58	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.E8257N.get_all_outputs() method), 60
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.HP_8648C.get_all_children() method), 59	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.HP_8642A.get_all_outputs() method), 56
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.HP_8648D.get_all_children() method), 60	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.HP_8643A.get_all_outputs() method), 55
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.RS_SMA100B.get_all_children() method), 54	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.HP_8648A.get_all_outputs() method), 57
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.RS_SMF100A.get_all_children() method), 53	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.HP_8648B.get_all_outputs() method), 58
get_all_children()	(naqs- lab_devices.SignalGenerator.Models.RS_SMHU.get_all_children() method), 55	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.HP_8648C.get_all_outputs() method), 59
get_all_children()	(naqs- lab_devices.SR865.labscript_device.SR865.get_all_children() method), 99	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.HP_8648D.get_all_outputs() method), 60
get_all_children()	(naqs- lab_devices.StaticFreqAmp method), 10	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.RS_SMA100B.get_all_outputs() method), 54
get_all_children()	(naqs- lab_devices.TektronixTDS.labscript_device.TDS_Scope.get_all_children() method), 104	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.RS_SMF100A.get_all_outputs() method), 53
get_all_children()	(naqs- lab_devices.VISA.labscript_device.VISA.get_all_children() method), 15	get_all_outputs()	(naqs- lab_devices.SignalGenerator.Models.RS_SMHU.get_all_outputs() method), 55
get_all_outputs()	(naqs- lab_devices.CounterScopeChannel method), 9	get_all_outputs()	(naqs- lab_devices.SR865.labscript_device.SR865.get_all_outputs() method), 99
get_all_outputs()	(naqs- lab_devices.KeysightDCSupply.labscript_device.KeysightDCSupply.get_all_outputs() method), 110	get_all_outputs()	(naqs- lab_devices.StaticFreqAmp method), 10
get_all_outputs()	(naqs- lab_devices.KeysightXSeries.labscript_device.KeysightXSeries.get_all_outputs() method), 81	get_all_outputs()	(naqs- lab_devices.TektronixTDS.labscript_device.TDS_Scope.get_all_outputs() method), 104

```

get_all_outputs() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
lab_devices.VISA.labscrip_device.VISA method), 62
method), 15 get_all_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SR865.blacs_tab.SR865Tab
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab), 97
method), 107 get_all_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab), 102
method), 78 get_all_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.VISA.blacs_tab.VISATab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab), 12
method), 83 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 107
method), 85 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 78
method), 87 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 73 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 67 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab), 87
method), 17 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab), 73
method), 21 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 67
method), 25 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 17
method), 27 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 21
method), 30 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 25
method), 32 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab), 27
method), 50 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
method), 48 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
method), 36 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
method), 40 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 48
method), 44 get_builtin_save_data() (naqs-
get_all_save_data() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10

```


method), 36
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N method), 40
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 40
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight method), 48
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 44
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab method), 36
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 40
 get_builton_save_data() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 97
 get_builton_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 44
 get_builton_save_data() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 102
 get_builton_save_data() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 62
 get_builton_save_data() (naqslab_devices.VISA.blacs_tab.VISATab method), 12
 get_channel() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 97
 get_channel() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 107
 get_channel() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 102
 get_channel() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 78
 get_channel() (naqslab_devices.VISA.blacs_tab.VISATab method), 12
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 83
 get_channel() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab method), 107
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 85
 get_channel() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 78
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 88
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 83
 get_channel() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 73
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 83
 get_channel() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 67
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 88
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 17
 get_channel() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 73
 get_channel() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 67
 get_channel() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 88
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 21
 get_channel() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 67
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 25
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 18
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 21
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 30
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 25
 get_channel() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 32

```

(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
method), 28
get_child_from_connection_table() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
method), 30
get_child_from_connection_table() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
method), 32
get_child_from_connection_table() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
(naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab
method), 50
get_child_from_connection_table() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
(naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
method), 48
get_child_from_connection_table() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
(naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
method), 36
get_child_from_connection_table() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
(naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
method), 40
get_child_from_connection_table() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 44
get_child_from_connection_table() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
(naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab), 21
method), 62
get_child_from_connection_table() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
(naqslab_devices.SR865.blacs_tab.SR865Tab
method), 97
get_child_from_connection_table() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
(naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTabmethod), 28
method), 102
get_child_from_connection_table() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
(naqslab_devices.VISA.blacs_tab.VISATab
method), 12
get_default_unit_conversion_classes() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
(naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B), 32
method), 93
get_default_unit_conversion_classes() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab
(naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B), 40
method), 92
get_default_unit_conversion_classes() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
(naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A), 48
method), 94
get_direct_outputs() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
method), 75
get_direct_outputs() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
method), 70
get_flag_number() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
method), 75
get_flag_number() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
method), 70
get_front_panel_properties() (naqslab_devices.SR865.blacs_tab.SR865Tab
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab), 97

```

`get_front_panel_properties()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 44
`method`), 102

`get_front_panel_properties()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab` method), 62
`method`), 12

`get_front_panel_values()` (naqs-
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` method), 97
`method`), 107

`get_front_panel_values()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 102
`method`), 78

`get_front_panel_values()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab` method), 12
`method`), 83

`get_front_panel_values()` (naqs-
`lab_devices.CounterScopeChannel` method),
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` 9
`method`), 85

`get_front_panel_values()` (naqs-
`lab_devices.KeysightDCSupply.labscript_device.KeysightDCSu`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 110
`method`), 88

`get_front_panel_values()` (naqs-
`lab_devices.KeysightXSeries.labscript_device.KeysightXScope`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` method), 73
`method`), 73

`get_front_panel_values()` (naqs-
`lab_devices.NovaTechDDS.labscript_device.NovaTech409B`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` method), 67
`method`), 67

`get_front_panel_values()` (naqs-
`lab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` method), 92
`method`), 18

`get_front_panel_values()` (naqs-
`lab_devices.NovaTechDDS.labscript_device.NovaTech440A`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` method), 94
`method`), 21

`get_front_panel_values()` (naqs-
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.Pulse`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 75
`method`), 25

`get_front_panel_values()` (naqs-
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBl`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 70
`method`), 28

`get_front_panel_values()` (naqs-
`lab_devices.ScopeChannel` method), 8
`method`), 30

`get_front_panel_values()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` method), 65
`method`), 32

`get_front_panel_values()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` method), 61
`method`), 50

`get_front_panel_values()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGen.E8257NTab` method), 56
`method`), 48

`get_front_panel_values()` (naqs-
`lab_devices.SignalGenerator.BLACS.KeysightSigGen.KeysightSigGen`
`lab_devices.SignalGenerator.Models.HP_8643A` method), 56
`method`), 36

`get_front_panel_values()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab` method), 57
`method`), 40

`get_front_panel_values()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648FTab` method), 58
`method`), 40

```

get_properties() (naqslab_devices.SignalGenerator.Models.HP_8648A method), 94
get_property() (naqslab_devices.SignalGenerator.Models.HP_8648A method), 59
get_properties() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 73
get_property() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 60
get_properties() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 75
get_property() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 60
get_properties() (naqslab_devices.SignalGenerator.Models.RS_SMA100B method), 54
get_property() (naqslab_devices.SignalGenerator.Models.RS_SMA100B method), 54
get_properties() (naqslab_devices.SignalGenerator.Models.RS_SMF100A method), 67
get_property() (naqslab_devices.SignalGenerator.Models.RS_SMF100A method), 53
get_properties() (naqslab_devices.SignalGenerator.Models.RS_SMHU method), 70
get_property() (naqslab_devices.SignalGenerator.Models.RS_SMHU method), 55
get_properties() (naqslab_devices.SR865.labscrip_device.SR865 method), 100
get_property() (naqslab_devices.SR865.labscrip_device.SR865 method), 100
get_properties() (naqslab_devices.StaticFreqAmp method), 10
get_property() (naqslab_devices.StaticFreqAmp method), 10
get_properties() (naqslab_devices.TektronixTDS.labscrip_device.TDSScope method), 104
get_property() (naqslab_devices.TektronixTDS.labscrip_device.TDSScope method), 104
get_properties() (naqslab_devices.VISA.labscrip_device.VISA method), 15
get_property() (naqslab_devices.VISA.labscrip_device.VISA method), 15
get_property() (naqslab_devices.CounterScopeChannel method), 9
get_properties() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupply method), 107
get_property() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupply method), 107
get_properties() (naqslab_devices.KeysightDCSupply.labscrip_device.KeysightDCSupply method), 110
get_property() (naqslab_devices.KeysightDCSupply.labscrip_device.KeysightDCSupply method), 110
get_properties() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXSeries method), 78
get_property() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXSeries method), 78
get_properties() (naqslab_devices.KeysightXSeries.labscrip_device.KeysightXSeries method), 81
get_property() (naqslab_devices.KeysightXSeries.labscrip_device.KeysightXSeries method), 81
get_properties() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTechDDS method), 83
get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTechDDS method), 83
get_properties() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTechDDS method), 86
get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTechDDS method), 86
get_properties() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTechDDS method), 88
get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTechDDS method), 88
get_properties() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTechDDS method), 93
get_property() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTechDDS method), 93
get_properties() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTechDDS method), 92
get_property() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTechDDS method), 92
get_properties() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTechDDS method), 61
get_property() (naqslab_devices.NovaTechDDS.labscrip_device.NovaTechDDS method), 61

```


`lab_devices.SignalGenerator.Models.HP_8642A.get_save_data()` (naqs-
`method`), 56 `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`get_property()` (naqs-`method`), 88
`lab_devices.SignalGenerator.Models.HP_8643A.get_save_data()` (naqs-
`method`), 56 `lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster`
`get_property()` (naqs-`method`), 73
`lab_devices.SignalGenerator.Models.HP_8648A.get_save_data()` (naqs-
`method`), 57 `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES`
`get_property()` (naqs-`method`), 67
`lab_devices.SignalGenerator.Models.HP_8648B.get_save_data()` (naqs-
`method`), 58 `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`get_property()` (naqs-`method`), 18
`lab_devices.SignalGenerator.Models.HP_8648C.get_save_data()` (naqs-
`method`), 59 `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`get_property()` (naqs-`method`), 21
`lab_devices.SignalGenerator.Models.HP_8648D.get_save_data()` (naqs-
`method`), 60 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`get_property()` (naqs-`method`), 25
`lab_devices.SignalGenerator.Models.RS_SMA100B.get_save_data()` (naqs-
`method`), 54 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`get_property()` (naqs-`method`), 28
`lab_devices.SignalGenerator.Models.RS_SMF100A.get_save_data()` (naqs-
`method`), 53 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`get_property()` (naqs-`method`), 30
`lab_devices.SignalGenerator.Models.RS_SMHU.get_save_data()` (naqs-
`method`), 55 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`get_property()` (naqs-`method`), 32
`lab_devices.SR865.blacs_tab.SR865Tab.get_save_data()` (naqs-
`method`), 97 `lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`get_property()` (naqs-`method`), 50
`lab_devices.SR865.labscript_device.SR865.get_save_data()` (naqs-
`method`), 100 `lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`get_property()` (naqslab_devices.StaticFreqAmp`method`), 48
`method`), 10 `get_save_data()` (naqs-
`get_property()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `method`), 36
`method`), 102 `get_save_data()` (naqs-
`get_property()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100`
`lab_devices.TektronixTDS.labscript_device.TDS_Scope` `method`), 40
`method`), 104 `get_save_data()` (naqs-
`get_property()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.VISA.blacs_tab.VISATab` `method`), 44
`method`), 12 `get_save_data()` (naqs-
`get_property()` (naqs-`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.VISA.labscript_device.VISA` `method`), 63
`method`), 15 `get_save_data()` (naqs-
`get_save_data()` (naqs-`lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` `method`), 97
`method`), 107 `get_save_data()` (naqs-
`get_save_data()` (naqs-`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` `method`), 102
`method`), 78 `get_save_data()` (naqs-
`get_save_data()` (naqs-`lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` `method`), 12
`method`), 83 `get_tab_layout()` (naqs-
`get_save_data()` (naqs-`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTa`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` `method`), 107
`method`), 86 `get_tab_layout()` (naqs-


```

lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
method), 32
HP_8648C (class in naqs-
hide_error() (naqs- lab_devices.SignalGenerator.Models),
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8357NTab
method), 50
HP_8648CTab (class in naqs-
hide_error() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648),
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
method), 48
HP_8648D (class in naqs-
hide_error() (naqs- lab_devices.SignalGenerator.Models),
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
method), 36
HP_8648DTab (class in naqs-
hide_error() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648),
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
method), 40
HP_8648Worker (class in naqs-
hide_error() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648),
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 44
hide_error() (naqs-
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDC
method), 63
attribute), 106
hide_error() (naqs- ICON_BUSY (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScop
lab_devices.SR865.blacs_tab.SR865Tab attribute), 77
method), 97
ICON_BUSY (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B
hide_error() (naqs- attribute), 82
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab ICON_BUSY (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B
method), 102 attribute), 84
hide_error() (naqs- ICON_BUSY (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440AT
lab_devices.VISA.blacs_tab.VISATab attribute), 86
method), 12
ICON_BUSY (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.Pu
HP_8642A (class in naqs- attribute), 72
lab_devices.SignalGenerator.Models), ICON_BUSY (naqslab_devices.PulseBlasterESRPro300.blacs_tab.Pulse
56 attribute), 66
HP_8642ATab (class in naqs- ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_
lab_devices.SignalGenerator.BLACS.HP_8642A), attribute), 16
16 ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_
HP_8642AWorker (class in naqs- attribute), 20
lab_devices.SignalGenerator.BLACS.HP_8642A) ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8
19 attribute), 24
HP_8643A (class in naqs- ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8
lab_devices.SignalGenerator.Models), attribute), 26
55 ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8
HP_8643ATab (class in naqs- attribute), 29
lab_devices.SignalGenerator.BLACS.HP_8643A) ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8
20 attribute), 31
HP_8643AWorker (class in naqs- ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens
lab_devices.SignalGenerator.BLACS.HP_8643A), attribute), 49
22 ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens
HP_8648A (class in naqs- attribute), 46
lab_devices.SignalGenerator.Models), ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.R
57 attribute), 35
HP_8648ATab (class in naqs- ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.R
lab_devices.SignalGenerator.BLACS.HP_8648), attribute), 39
24 ICON_BUSY (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_S
HP_8648B (class in naqs- attribute), 43
lab_devices.SignalGenerator.Models), ICON_BUSY (naqslab_devices.SignalGenerator.blacs_tab.SignalGenera
58 attribute), 61
HP_8648BTab (class in naqs- ICON_BUSY (naqslab_devices.SR865.blacs_tab.SR865Tab
lab_devices.SignalGenerator.BLACS.HP_8648), attribute), 95

```

ICON_BUSY (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab attribute), 100
 ICON_BUSY (naqslab_devices.VISA.blacs_tab.VISATab attribute), 11
 ICON_ERROR (naqslab_devices.SR865.blacs_tab.SR865Tab attribute), 95
 ICON_ERROR (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab attribute), 106
 ICON_ERROR (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab attribute), 100
 ICON_ERROR (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab attribute), 77
 ICON_ERROR (naqslab_devices.VISA.blacs_tab.VISATab attribute), 11
 ICON_ERROR (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab attribute), 82
 ICON_ERROR (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab attribute), 106
 ICON_ERROR (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab attribute), 84
 ICON_ERROR (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab attribute), 77
 ICON_ERROR (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab attribute), 86
 ICON_ERROR (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab attribute), 82
 ICON_ERROR (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab attribute), 72
 ICON_ERROR (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab attribute), 84
 ICON_ERROR (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab attribute), 66
 ICON_ERROR (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab attribute), 86
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab attribute), 16
 ICON_ERROR (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab attribute), 72
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab attribute), 20
 ICON_ERROR (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab attribute), 66
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab attribute), 24
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab attribute), 16
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab attribute), 26
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab attribute), 20
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab attribute), 24
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab attribute), 26
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab attribute), 20
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab attribute), 31
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab attribute), 26
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab attribute), 49
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab attribute), 46
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab attribute), 35
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab attribute), 49
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab attribute), 39
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab attribute), 39
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab attribute), 47
 ICON_ERROR (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab attribute), 43

lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab attribute), 35	(naqslab_devices.SMA100BTab attribute), 96
ICON_FATAL_ERROR (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab attribute), 39	ICON_OK (naqslab_devices.VISA.blacs_tab.VISATab attribute), 11
ICON_FATAL_ERROR (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab attribute), 43	(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab attribute), 108
ICON_FATAL_ERROR (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab attribute), 61	(naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyTab attribute), 108
ICON_FATAL_ERROR (naqslab_devices.SR865.blacs_tab.SR865Tab attribute), 96	init () (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeTab method), 79
ICON_FATAL_ERROR (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab attribute), 101	init () (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_A60Tab method), 89
ICON_FATAL_ERROR (naqslab_devices.VISA.blacs_tab.VISATab attribute), 11	init () (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BW_A60Tab method), 90
ICON_OK (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab attribute), 106	init () (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AW_A60Tab method), 90
ICON_OK (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab attribute), 77	init () (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Tab method), 69
ICON_OK (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_A60Tab attribute), 82	init () (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Tab method), 19
ICON_OK (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BW_A60Tab attribute), 84	init () (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 22
ICON_OK (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440AW_A60Tab attribute), 86	init () (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 33
ICON_OK (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab attribute), 72	init () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648VTab method), 33
ICON_OK (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab attribute), 66	init () (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab method), 51
ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab attribute), 16	init () (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab method), 16
ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab attribute), 20	init () (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 20
ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648VTab attribute), 24	init () (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorTab method), 24
ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648VTab attribute), 26	init () (naqslab_devices.SR865.blacs_worker.SR865Worker method), 26
ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648VTab attribute), 29	init () (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker method), 103
ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648VTab attribute), 31	init () (naqslab_devices.VISA.blacs_worker.VISAWorker method), 13
ICON_OK (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab attribute), 49	init_device_group () (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab method), 9
ICON_OK (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab attribute), 35	init_device_group () (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 81
ICON_OK (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab attribute), 39	init_device_group () (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 43
ICON_OK (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab attribute), 43	init_device_group () (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_A60Tab method), 93
ICON_OK (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab attribute), 61	init_device_group () (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409BW_A60Tab method), 93

```

lab_devices.NovaTechDDS.labscript_device.NovaTech440A.blacs_tab.KeysightDCSupply.blacs_worker.KeysightDCSupply
method), 92                                     attribute), 108
init_device_group() (naqs- initialise_GUI() (naqs-
lab_devices.NovaTechDDS.labscript_device.NovaTech440A.blacs_tab.KeysightDCSupplyTab
method), 94                                     method), 106
init_device_group() (naqs- initialise_GUI() (naqs-
lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster.NoiseDDS_200.blacs_tab.KeysightXScopeTab
method), 75                                     method), 77
init_device_group() (naqs- initialise_GUI() (naqs-
lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300.blacs_tab.NovaTech409B_ACTab
method), 70                                     method), 82
init_device_group() (naqs- initialise_GUI() (naqs-
lab_devices.ScopeChannel method), 8 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
init_device_group() (naqs- method), 86
lab_devices.SignalGenerator.labscript_device.SignalGenerator initialise_GUI() (naqs-
method), 65 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
init_device_group() (naqs- method), 86
lab_devices.SignalGenerator.Models.E8257N initialise_GUI() (naqs-
method), 61 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
init_device_group() (naqs- method), 73
lab_devices.SignalGenerator.Models.HP_8642A initialise_GUI() (naqs-
method), 56 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
init_device_group() (naqs- method), 68
lab_devices.SignalGenerator.Models.HP_8643A initialise_GUI() (naqs-
method), 56 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
init_device_group() (naqs- method), 18
lab_devices.SignalGenerator.Models.HP_8648A initialise_GUI() (naqs-
method), 57 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
init_device_group() (naqs- method), 22
lab_devices.SignalGenerator.Models.HP_8648B initialise_GUI() (naqs-
method), 58 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
init_device_group() (naqs- method), 25
lab_devices.SignalGenerator.Models.HP_8648C initialise_GUI() (naqs-
method), 59 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
init_device_group() (naqs- method), 28
lab_devices.SignalGenerator.Models.HP_8648D initialise_GUI() (naqs-
method), 60 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
init_device_group() (naqs- method), 30
lab_devices.SignalGenerator.Models.RS_SMA100B initialise_GUI() (naqs-
method), 54 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
init_device_group() (naqs- method), 33
lab_devices.SignalGenerator.Models.RS_SMF100A initialise_GUI() (naqs-
method), 53 lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
init_device_group() (naqs- method), 50
lab_devices.SignalGenerator.Models.RS_SMHU initialise_GUI() (naqs-
method), 55 lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
init_device_group() (naqs- method), 48
lab_devices.SR865.labscript_device.SR865 initialise_GUI() (naqs-
method), 100 lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10
init_device_group() (naqs- method), 36
lab_devices.StaticFreqAmp method), 10 initialise_GUI() (naqs-
init_device_group() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10
lab_devices.TektronixTDS.labscript_device.TDS_Scope method), 40
method), 104 initialise_GUI() (naqs-
init_device_group() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
lab_devices.VISA.labscript_device.VISA method), 44
method), 15 initialise_GUI() (naqs-
init_string (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab

```

Index	159
--------------	------------

lab_devices.SignalGenerator.BLACS.RS_SMHU100.SMHU100Worker (naqs-
method), 46 lab_devices.PulseBlasterESRPro300.runviewer_parser.PulseBl
interrupt_startup() (naqs- attribute), 71
lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
method), 64

M

interrupt_startup() (naqs- mainloop() (naqs-
lab_devices.SR865.blacs_worker.SR865Worker lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTa
method), 99 method), 107
interrupt_startup() (naqs- mainloop() (naqs-
lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorktab_devices.KeysightDCSupply.blacs_worker.KeysightDCSuppl
method), 103 method), 109
interrupt_startup() (naqs- mainloop() (naqs-
lab_devices.VISA.blacs_worker:VISAWorker lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
method), 14 method), 78
is_master_pseudoclock() (naqs- mainloop() (naqs-
lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200.blacs_worker.KeysightXScopeWor
property), 75 method), 80
is_master_pseudoclock() (naqs- mainloop() (naqs-
lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300.blacs_tab.NovaTech409B_ACTab
property), 70 method), 84

K

mainloop() (naqs-
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
method), 86
KeysightDCSupply (class in naqs-
lab_devices.KeysightDCSupply.labscript_device), mainloop() (naqs-
110 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
method), 88
KeysightDCSupplyTab (class in naqs-
lab_devices.KeysightDCSupply.blacs_tab), mainloop() (naqs-
105 lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACW
method), 89
KeysightDCSupplyWorker (class in naqs-
lab_devices.KeysightDCSupply.blacs_worker), mainloop() (naqs-
108 lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker
method), 90
KeysightSigGenTab (class in naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGen), mainloop() (naqs-
46 lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker
method), 91
KeysightSigGenWorker (class in naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGen), mainloop() (naqs-
51 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
method), 73
KeysightXScope (class in naqs-
lab_devices.KeysightXSeries.labscript_device), mainloop() (naqs-
81 lab_devices.PulseBlaster_No_DDS_200.blacs_worker.Pulsebla
method), 74
KeysightXScopeTab (class in naqs-
lab_devices.KeysightXSeries.blacs_tab), mainloop() (naqs-
77 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
method), 68
KeysightXScopeWorker (class in naqs-
lab_devices.KeysightXSeries.blacs_worker), mainloop() (naqs-
79 lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlaste
method), 69

L

mainloop() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
method), 88
labscript_device_class_name (naqs-
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
attribute), 73 mainloop() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor
labscript_device_class_name (naqs-
lab_devices.PulseBlaster_No_DDS_200.runviewer_parser.PulseBlaster_No_DDS_200_Parser
attribute), 76 mainloop() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
labscript_device_class_name (naqs-
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
attribute), 68 mainloop() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor


```

        method), 23
mainloop() (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 103
        method), 25
mainloop() (naqslab_devices.VISA.blacs_tab.VISATab
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 12
        method), 28
mainloop() (naqslab_devices.VISA.blacs_worker.VISAWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 14
        method), 30
mainloop() merge_registers() (naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupply
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 109
        method), 33
mainloop() minimum_recovery_time (naqslab_devices.KeysightXSeries.labscrip_device.KeysightXScope
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab), 81
        method), 34
mainloop() minimum_recovery_time (naqslab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab), 76
        method), 50
mainloop() minimum_recovery_time (naqslab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlaster
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab), 78
        method), 48
mainloop() minimum_recovery_time (naqslab_devices.TektronixTDS.labscrip_device.TDS_Scope
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker
        method), 52
mainloop() mode() (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupply
        lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab), 107
        method), 36
mainloop() mode() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
        lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker
        method), 38
mainloop() mode() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACT
        lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker
        method), 38
mainloop() mode() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab), 86
        method), 40
mainloop() mode() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
        method), 42
mainloop() mode() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
        method), 42
mainloop() mode() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlaster
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 68
        method), 44
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker), 22
        method), 46
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648A
        lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab), 25
        method), 63
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648B
        lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker), 28
        method), 64
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648C
        lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab), 30
        method), 33
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648D
        lab_devices.SR865.blacs_tab.SR865Tab), 33
        method), 97
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E82
        lab_devices.SR865.blacs_worker.SR865Worker), 50
        method), 99
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.Key
        lab_devices.SR865.blacs_worker.SR865Worker), 48
        method), 36
mainloop() mode() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA
        lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab), 40
        method), 102

```

mode () (*naqslab_devices.SignalGenerator.BLACS.RS_SMHU.Tab* *naqslab_devices.PulseBlaster_No_DDS_200.labscript_device*), 44 (module), 75

mode () (*naqslab_devices.SignalGenerator.blacs_tab.SignalGenerator.Tab* *naqslab_devices.PulseBlaster_No_DDS_200.register_classes*), 63 (module), 76

mode () (*naqslab_devices.SR865.blacs_tab.SR865Tab* *naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser*), 97 (module), 76

mode () (*naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab* *naqslab_devices.PulseBlasterESRPro300*), 102 (module), 66

mode () (*naqslab_devices.VISA.blacs_tab.VISATab* *naqslab_devices.PulseBlasterESRPro300.blacs_tab*), 12 (module), 66

model_ident (*naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker*), 69

naqslab_devices (*naqslab_devices.PulseBlasterESRPro300.blacs_worker*), 7

naqslab_devices.KeysightDCSupply (*naqslab_devices.PulseBlasterESRPro300.labscript_device*), 105 (module), 71

naqslab_devices.KeysightDCSupply.blacs_tab (*naqslab_devices.PulseBlasterESRPro300.runviewer_parser*), 105 (module), 24

naqslab_devices.KeysightDCSupply.blacs_worker (*naqslab_devices.SignalGenerator.BLACS.HP_8642A*), 108 (module), 46

naqslab_devices.KeysightDCSupply.labscript_device (*naqslab_devices.SignalGenerator.BLACS.HP_8643A*), 110 (module), 35

naqslab_devices.KeysightDCSupply.register_classes (*naqslab_devices.SignalGenerator.BLACS.HP_8648*), 111 (module), 39

naqslab_devices.KeysightXSeries (*naqslab_devices.SignalGenerator.BLACS.KeysightSignalGenerator*), 76 (module), 42

naqslab_devices.KeysightXSeries.blacs_tab (*naqslab_devices.SignalGenerator.BLACS.RS_SMA100B*), 77 (module), 61

naqslab_devices.KeysightXSeries.blacs_worker (*naqslab_devices.SignalGenerator.BLACS.RS_SMF100A*), 79 (module), 63

naqslab_devices.KeysightXSeries.labscript_device (*naqslab_devices.SignalGenerator.BLACS.RS_SMHU*), 81 (module), 65

naqslab_devices.KeysightXSeries.register_classes (*naqslab_devices.SignalGenerator.blacs_tab*), 82 (module), 53

naqslab_devices.NovaTechDDS (*naqslab_devices.SignalGenerator.blacs_worker*), 82 (module), 66

naqslab_devices.NovaTechDDS.blacs_tab (*naqslab_devices.SignalGenerator.blacs_worker*), 82 (module), 95

naqslab_devices.NovaTechDDS.blacs_worker (*naqslab_devices.SignalGenerator.blacs_worker*), 89 (module), 98

naqslab_devices.NovaTechDDS.labscript_device (*naqslab_devices.SignalGenerator.blacs_worker*), 91 (module), 99

naqslab_devices.NovaTechDDS.register_classes (*naqslab_devices.SignalGenerator.labscript_device*), 95 (module), 100

naqslab_devices.NovaTechDDS.runviewer_parser (*naqslab_devices.SignalGenerator.Models*), 95 (module), 100

naqslab_devices.PulseBlaster_No_DDS_200 (*naqslab_devices.SignalGenerator.register_classes*), 71 (module), 100

naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab (*naqslab_devices.SR865*), 72 (module), 100

naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker (*naqslab_devices.SR865.blacs_tab*), 74 (module), 103

naqslab_devices.TektronixTDS (*naqslab_devices.SR865.blacs_worker*), 100 (module), 100

naqslab_devices.TektronixTDS.blacs_tab (*naqslab_devices.SR865.labscript_device*), 100 (module), 100

naqslab_devices.TektronixTDS.blacs_worker (*naqslab_devices.SR865.register_classes*), 103 (module), 103

naqslab_devices.TektronixTDS.labscript_device (*naqslab_devices.SR865.runviewer_parser*), 103 (module), 103

(*module*), 104

naqslab_devices.TektronixTDS.register_classes(*attribute*), 71

(*module*), 105

naqslab_devices.VISA(*module*), 10

naqslab_devices.VISA.blacs_tab(*module*), 11

naqslab_devices.VISA.blacs_worker(*module*), 13

naqslab_devices.VISA.labscript_device(*module*), 14

naqslab_devices.VISA.register_classes(*module*), 15

NovaTech409B(*class in naqslab_devices.NovaTechDDS.labscript_device*), 92

NovaTech409B_AC(*class in naqslab_devices.NovaTechDDS.labscript_device*), 91

NovaTech409B_ACParser(*class in naqslab_devices.NovaTechDDS.runviewer_parser*), 95

NovaTech409B_ACTab(*class in naqslab_devices.NovaTechDDS.blacs_tab*), 82

NovaTech409B_ACWorker(*class in naqslab_devices.NovaTechDDS.blacs_worker*), 89

NovaTech409BParser(*class in naqslab_devices.NovaTechDDS.runviewer_parser*), 95

NovaTech409BTab(*class in naqslab_devices.NovaTechDDS.blacs_tab*), 84

NovaTech409BWorker(*class in naqslab_devices.NovaTechDDS.blacs_worker*), 89

NovaTech440A(*class in naqslab_devices.NovaTechDDS.labscript_device*), 93

NovaTech440AParser(*class in naqslab_devices.NovaTechDDS.runviewer_parser*), 95

NovaTech440ATab(*class in naqslab_devices.NovaTechDDS.blacs_tab*), 86

NovaTech440AWorker(*class in naqslab_devices.NovaTechDDS.blacs_worker*), 90

num_dds(*naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser.attribute*), 76

num_dds(*naqslab_devices.PulseBlasterESRPro300.runviewer_parser.attribute*), 71

num_DO(*naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.attribute*), 72

num_DO(*naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab.attribute*), 66

num_flags(*naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser.attribute*), 76

num_flags(*naqslab_devices.PulseBlasterESRPro300.runviewer_parser.attribute*), 71

O

offset_instructions_from_trigger()
(*naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.method*), 76

offset_instructions_from_trigger()
(*naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300Tab.method*), 70

on_force_full_buffered_reprogram()
(*naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab.method*), 107

on_force_full_buffered_reprogram()
(*naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab.method*), 78

on_force_full_buffered_reprogram()
(*naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_AC.method*), 84

on_force_full_buffered_reprogram()
(*naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab.method*), 86

on_force_full_buffered_reprogram()
(*naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab.method*), 88

on_force_full_buffered_reprogram()
(*naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab.method*), 73

on_force_full_buffered_reprogram()
(*naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab.method*), 68

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642A.method*), 18

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A.method*), 22

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648A.method*), 26

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648B.method*), 28

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648C.method*), 30

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648D.method*), 35

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8363B.method*), 50

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8363C.method*), 48

on_force_full_buffered_reprogram()
(*naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100B.method*), 37

on_force_full_buffered_reprogram() *lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N*
 (*naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab*
 method), 40 on_resolve_value_inconsistency() (*naqs-*
 on_force_full_buffered_reprogram() *lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight*
 (*naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab*
 method), 44 on_resolve_value_inconsistency() (*naqs-*
 on_force_full_buffered_reprogram() *lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100*
 (*naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab*), 37
 method), 63 on_resolve_value_inconsistency() (*naqs-*
 on_force_full_buffered_reprogram() *lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100*
 (*naqslab_devices.SR865.blacs_tab.SR865Tab* method), 40
 method), 97 on_resolve_value_inconsistency() (*naqs-*
 on_force_full_buffered_reprogram() *lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab*
 (*naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab*method), 44
 method), 102 on_resolve_value_inconsistency() (*naqs-*
 on_force_full_buffered_reprogram() *lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab*
 (*naqslab_devices.VISA.blacs_tab.VISATab* method), 63
 method), 12 on_resolve_value_inconsistency() (*naqs-*
 on_resolve_value_inconsistency() (*naqs-* *lab_devices.SR865.blacs_tab.SR865Tab*
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab), 97
 method), 107 on_resolve_value_inconsistency() (*naqs-*
 on_resolve_value_inconsistency() (*naqs-* *lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab*
*lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab*method), 102
 method), 78 on_resolve_value_inconsistency()
 on_resolve_value_inconsistency() (*naqs-* (*naqslab_devices.VISA.blacs_tab.VISATab*
*lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab*method), 12
 method), 84
 on_resolve_value_inconsistency() (*naqs-* **P**
*lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab*parent_clock_line() (*naqs-*
 method), 86 *lab_devices.CounterScopeChannel* prop-
 on_resolve_value_inconsistency() (*naqs-* erty), 9
*lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab*parent_clock_line() (*naqs-*
 method), 88 *lab_devices.KeysightDCSupply.labscrip_device.KeysightDCSu*
 on_resolve_value_inconsistency() (*naqs-* property), 110
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab (*naqs-*
 method), 73 *lab_devices.KeysightXSeries.labscrip_device.KeysightXScope*
 on_resolve_value_inconsistency() (*naqs-* property), 81
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab (*naqs-*
 method), 68 *lab_devices.NovaTechDDS.labscrip_device.NovaTech409B*
 on_resolve_value_inconsistency() (*naqs-* property), 93
*lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab*parent_clock_line() (*naqs-*
 method), 18 *lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_AC*
 on_resolve_value_inconsistency() (*naqs-* property), 92
*lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab*parent_clock_line() (*naqs-*
 method), 22 *lab_devices.NovaTechDDS.labscrip_device.NovaTech440A*
 on_resolve_value_inconsistency() (*naqs-* property), 94
*lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab*parent_clock_line() (*naqs-*
 method), 26 *lab_devices.PulseBlaster_No_DDS_200.labscrip_device.Pulse*
 on_resolve_value_inconsistency() (*naqs-* property), 76
*lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab*parent_clock_line() (*naqs-*
 method), 28 *lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBl*
 on_resolve_value_inconsistency() (*naqs-* property), 70
*lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab*parent_clock_line() (*naqs-*
 method), 30 *lab_devices.ScopeChannel* property), 8
 on_resolve_value_inconsistency() (*naqs-* parent_clock_line() (*naqs-*
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab *lab_devices.SignalGenerator.labscrip_device.SignalGenerator*
 method), 33 property), 65
 on_resolve_value_inconsistency() (*naqs-* parent_clock_line() (*naqs-*

`lab_devices.SignalGenerator.Models.E8257N` `primary_worker()` `(naqs-`
`property), 61` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`parent_clock_line()` `(naqs-` `property), 84`
`lab_devices.SignalGenerator.Models.HP_8642A` `primary_worker()` `(naqs-`
`property), 56` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`parent_clock_line()` `(naqs-` `property), 86`
`lab_devices.SignalGenerator.Models.HP_8643A` `primary_worker()` `(naqs-`
`property), 56` `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`parent_clock_line()` `(naqs-` `property), 88`
`lab_devices.SignalGenerator.Models.HP_8648A` `primary_worker()` `(naqs-`
`property), 57` `lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster`
`parent_clock_line()` `(naqs-` `property), 73`
`lab_devices.SignalGenerator.Models.HP_8648B` `primary_worker()` `(naqs-`
`property), 58` `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES`
`parent_clock_line()` `(naqs-` `property), 68`
`lab_devices.SignalGenerator.Models.HP_8648C` `primary_worker()` `(naqs-`
`property), 59` `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`parent_clock_line()` `(naqs-` `property), 18`
`lab_devices.SignalGenerator.Models.HP_8648D` `primary_worker()` `(naqs-`
`property), 60` `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`parent_clock_line()` `(naqs-` `property), 22`
`lab_devices.SignalGenerator.Models.RS_SMA100B` `primary_worker()` `(naqs-`
`property), 54` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`parent_clock_line()` `(naqs-` `property), 26`
`lab_devices.SignalGenerator.Models.RS_SMF100A` `primary_worker()` `(naqs-`
`property), 53` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`parent_clock_line()` `(naqs-` `property), 28`
`lab_devices.SignalGenerator.Models.RS_SMHU` `primary_worker()` `(naqs-`
`property), 55` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`parent_clock_line()` `(naqs-` `property), 30`
`lab_devices.SR865.labscrip_device.SR865` `primary_worker()` `(naqs-`
`property), 100` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`parent_clock_line()` `(naqs-` `property), 33`
`lab_devices.StaticFreqAmp` `property), primary_worker()` `(naqs-`
`10` `lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`parent_clock_line()` `(naqs-` `property), 50`
`lab_devices.TektronixTDS.labscrip_device.TDS_Scope` `primary_worker()` `(naqs-`
`property), 105` `lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`parent_clock_line()` `(naqs-` `property), 48`
`lab_devices.VISA.labscrip_device.VISA` `primary_worker()` `(naqs-`
`property), 15` `lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10`
`pb_instructions` `(naqs-` `property), 37`
`lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200` `(naqs-`
`attribute), 76` `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10`
`pb_instructions` `(naqs-` `property), 40`
`lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300` `(naqs-`
`attribute), 70` `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`phase (naqslab_devices.SR865.labscrip_device.SR865` `property), 44`
`attribute), 99` `primary_worker()` `(naqs-`
`phase_parser()` `(naqs-` `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.SR865.blacs_worker.SR865Worker` `property), 63`
`method), 98` `primary_worker()` `(naqs-`
`primary_worker()` `(naqs-` `lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` `property), 97`
`property), 107` `primary_worker()` `(naqs-`
`primary_worker()` `(naqs-` `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` `property), 102`
`property), 78` `primary_worker()` `(naqs-`

`lab_devices.VISA.blacs_tab.VISATab` `property`), 12
`program_device()` (`naqs-` `lab_devices.SR865.blacs_tab.SR865Tab` `method`), 97
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` `method`), 107
`program_device()` (`naqs-` `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `method`), 102
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` `method`), 78
`program_device()` (`naqs-` `lab_devices.VISA.blacs_tab.VISATab` `method`), 12
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` `method`), 84
`program_device()` (`naqs-` `lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` `method`), 107
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` `method`), 86
`program_device()` (`naqs-` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` `method`), 78
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` `method`), 88
`program_device()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` `method`), 84
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` `method`), 73
`program_device()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` `method`), 86
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` `method`), 68
`program_device()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` `method`), 88
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` `method`), 18
`program_device()` (`naqs-` `lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` `method`), 73
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` `method`), 22
`program_device()` (`naqs-` `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` `method`), 68
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` `method`), 26
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` `method`), 18
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` `method`), 28
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` `method`), 22
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` `method`), 30
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` `method`), 26
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` `method`), 33
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` `method`), 28
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab` `method`), 50
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` `method`), 48
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab` `method`), 48
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` `method`), 37
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab` `method`), 37
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab` `method`), 40
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` `method`), 40
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab` `method`), 48
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` `method`), 44
`program_device()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab` `method`), 37
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` `method`), 37

```

program_device_properties() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
method), 40 program_manual() (naqs-
program_device_properties() (naqs- lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
method), 44 program_manual() (naqs-
program_device_properties() (naqs- lab_devices.SR865.blacs_worker.SR865Worker
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
method), 63 program_manual() (naqs-
program_device_properties() (naqs- lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
lab_devices.SR865.blacs_tab.SR865Tab
method), 97 program_manual() (naqs-
program_device_properties() (naqs- lab_devices.VISA.blacs_worker.VISAWorker
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 102 program_manual() (naqs-
program_device_properties() (naqs- lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker
lab_devices.VISA.blacs_tab.VISATab
method), 12 program_manual() (naqs-
program_manual() (naqs- lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker
lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker
method), 108 program_manual() (naqs-
program_manual() (naqs- lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker
lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
method), 80 program_manual() (naqs-
program_manual() (naqs- lab_devices.SR865.blacs_worker.SR865Worker
lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker
method), 89 program_manual() (naqs-
program_manual() (naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlasterNoDDS200Worker
lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker
method), 90 program_manual() (naqs-
program_manual() (naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300Worker
lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker
method), 91 program_manual() (naqs-
program_manual() (naqs- lab_devices.CounterScopeChannel
lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlasterNoDDS200Worker
method), 74 program_manual() (naqs-
program_manual() (naqs- lab_devices.KeysightDCSupply.labscript_device.KeysightDCSupplyWorker
lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker
method), 69 program_manual() (naqs-
program_manual() (naqs- lab_devices.KeysightXSeries.labscript_device.KeysightXScopeWorker
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
method), 19 program_manual() (naqs-
program_manual() (naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B_ACWorker
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
method), 23 program_manual() (naqs-
program_manual() (naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B_ACWorker
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker
method), 34 program_manual() (naqs-
program_manual() (naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech440AWorker
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensWorker
method), 52 program_manual() (naqs-
program_manual() (naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlasterNoDDS200Worker
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker
method), 38 program_manual() (naqs-
program_manual() (naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300Worker
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
method), 42 program_manual() (naqs-
program_manual() (naqs- lab_devices.ScopeChannel property), 8

```


169

Method	Module	Module Path	Module Name
queue_work()	lab_devices.NovaTechDDS.blacs_tab.NovaTech440BTab	read_analog_parameters_string	(naqs-lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker.attribute), 79
queue_work()	lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab	read_counter_string	(naqs-lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker.attribute), 79
queue_work()	lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab	read_analog_parameters_string	(naqs-lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker.attribute), 79
queue_work()	lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab	read_parser()	(naqs-lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker.method), 108
queue_work()	lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab	read_string	(naqs-lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker.attribute), 108
queue_work()	lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab	read_string	(naqs-lab_devices.SR865.blacs_worker.SR865Worker.attribute), 98
queue_work()	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab	read_waveform_string	(naqs-lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker.attribute), 79
queue_work()	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab	read_waveform_string	(naqs-lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker.attribute), 103
queue_work()	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab	read_xy_parameters_string	(naqs-lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker.attribute), 103
queue_work()	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab	read_xy_parameters_string	(naqs-lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker.attribute), 103
queue_work()	lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab	reset()	(naqs-lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab.method), 73
queue_work()	lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab	reset()	(naqs-lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab.method), 48
queue_work()	lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab	restart()	(naqs-lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyWorker.method), 107
queue_work()	lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab	restart()	(naqs-lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeWorker.attribute), 79

`method), 78`
`restart () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method), 84`
`restart () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab, 68`
`method), 86`
`restart () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method), 88`
`restart () (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method), 73`
`restart () (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method), 68`
`restart () (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`method), 18`
`restart () (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`method), 22`
`restart () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`method), 26`
`restart () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`method), 28`
`restart () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`method), 30`
`restart () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`method), 33`
`restart () (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`method), 51`
`restart () (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab`
`method), 48`
`restart () (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`method), 37`
`restart () (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method), 40`
`restart () (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method), 44`
`restart () (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method), 63`
`restart () (naqslab_devices.SR865.blacs_tab.SR865Tab`
`method), 97`
`restart () (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method), 102`
`restart () (naqslab_devices.VISA.blacs_tab.VISATab`
`method), 13`
`restore_built_in_save_data () (naqslab_devices.SR865.blacs_tab.SR865Tab`
`method), 107`
`restore_built_in_save_data () (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method), 79`
`restore_built_in_save_data () (naqslab_devices.VISA.blacs_tab.VISATab`
`method), 13`
`restore_built_in_save_data () (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method), 107`
`restore_built_in_save_data () (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method), 86`
`restore_built_in_save_data () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method), 84`
`restore_built_in_save_data () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method), 88`
`restore_built_in_save_data () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`

```

restore_save_data() (naqslab_devices.SignalGenerator.Models),
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab 53
method), 86 RS_SMA100BTab (class in naqslab_devices.SignalGenerator.BLACS.RS_SMA100B),
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B),
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab 35
method), 88 RS_SMA100BWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMA100B),
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B),
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 73 RS_SMF100A (class in naqslab_devices.SignalGenerator.Models),
restore_save_data() (naqslab_devices.SignalGenerator.Models),
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 68 RS_SMF100ATab (class in naqslab_devices.SignalGenerator.BLACS.RS_SMF100A),
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A),
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
method), 18 RS_SMF100AWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMF100A),
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A),
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
method), 22 RS_SMHU (class in naqslab_devices.SignalGenerator.Models),
restore_save_data() (naqslab_devices.SignalGenerator.Models),
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
method), 26 RS_SMHUTab (class in naqslab_devices.SignalGenerator.BLACS.RS_SMHU),
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU),
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
method), 28 RS_SMHUWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMHU),
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU),
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
method), 30 run() (naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker
restore_save_data() (naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker
method), 109 method), 109
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
method), 33 method), 80
restore_save_data() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_AutoWorker
lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257MTab 89
method), 51 run() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_AutoWorker
restore_save_data() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_AutoWorker
method), 90 method), 90
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab NovaTechDDS.blacs_worker.NovaTech440AWorker
method), 48 method), 91
restore_save_data() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200_Tab
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
method), 37 run() (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300_Tab
restore_save_data() (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300_Tab
method), 69 method), 69
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
method), 41 method), 20
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 23 method), 23
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648WTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648WTab
method), 44 run() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648WTab
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648WTab
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab
method), 63 method), 52
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
lab_devices.SR865.blacs_tab.SR865Tab
method), 38 method), 38
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 42 method), 42
restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 102 method), 46
restore_save_data() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
lab_devices.VISA.blacs_tab.VISATab
method), 65 method), 65
RS_SMA100B (class in naqslab_devices.SR865.blacs_worker.SR865Worker
method), 99 method), 99

```



```
run () (naqslab_devices.TektronixTDS.blacs_worker.TDSScopeChannel, 53)
      (method), 103
run () (naqslab_devices.VISA.blacs_worker.VISAWorker.scale_factor, 14)
      (lab_devices.SignalGenerator.Models.RS_SMHU
       attribute), 54

S
ScopeChannel (class in naqslab_devices), 7
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
attribute), 19) (lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
                  (method), 107)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
attribute), 22) (lab_devices.KeysightXSeries.blacs_tab.KeysightXSpectrumAnalyzerTab
                  (method), 79)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
attribute), 33) (lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
                  (method), 18)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker
attribute), 52) (lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
                  (method), 22)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker
attribute), 37) (lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
                  (method), 26)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
attribute), 41) (lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB
                  (method), 28)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
attribute), 45) (lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
                  (method), 30)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.blacs_worker.SignalGeneratorTab
attribute), 63) (lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB
                  (method), 33)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.labscript_device.SignalGeneratorTab
attribute), 65) (lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
                  (method), 51)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.E8257N
attribute), 60) (lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenWorker
                  (method), 48)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.HP_8642A
attribute), 56) (lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker
                  (method), 37)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.HP_8643A
attribute), 55) (lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
                  (method), 41)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.HP_8648A
attribute), 57) (lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
                  (method), 44)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.HP_8648B
attribute), 58) (lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
                  (method), 63)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.HP_8648C
attribute), 59) (lab_devices.SR865.blacs_tab.SR865Tab
                  (method), 97)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.HP_8648D
attribute), 59) (lab_devices.TektronixTDS.blacs_tab.TDSScopeTab
                  (method), 102)
scale_factor (naqs- send_clear() (naqs-
lab_devices.SignalGenerator.Models.RS_SMA100B
attribute), 54) (lab_devices.VISA.blacs_tab.VISATab
                  (method), 11)
scale_factor (naqs- sens (naqslab_devices.SR865.labscript_device.SR865
```


`lab_devices.SignalGenerator.Models.HP_8642A.set_tab_icon_and_colour()` (naqs-
`method`), 57 `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`set_property()` (naqs-`method`), 68
`lab_devices.SignalGenerator.Models.HP_8643A.set_tab_icon_and_colour()` (naqs-
`method`), 56 `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`set_property()` (naqs-`method`), 18
`lab_devices.SignalGenerator.Models.HP_8648A.set_tab_icon_and_colour()` (naqs-
`method`), 58 `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`set_property()` (naqs-`method`), 22
`lab_devices.SignalGenerator.Models.HP_8648B.set_tab_icon_and_colour()` (naqs-
`method`), 58 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`set_property()` (naqs-`method`), 26
`lab_devices.SignalGenerator.Models.HP_8648C.set_tab_icon_and_colour()` (naqs-
`method`), 59 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`set_property()` (naqs-`method`), 28
`lab_devices.SignalGenerator.Models.HP_8648D.set_tab_icon_and_colour()` (naqs-
`method`), 60 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`set_property()` (naqs-`method`), 31
`lab_devices.SignalGenerator.Models.RS_SMA100B.set_tab_icon_and_colour()` (naqs-
`method`), 54 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`set_property()` (naqs-`method`), 33
`lab_devices.SignalGenerator.Models.RS_SMF100A.set_tab_icon_and_colour()` (naqs-
`method`), 53 `lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`set_property()` (naqs-`method`), 51
`lab_devices.SignalGenerator.Models.RS_SMHU.set_tab_icon_and_colour()` (naqs-
`method`), 55 `lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGens`
`set_property()` (naqs-`method`), 48
`lab_devices.SR865.labscrip_device.SR865.set_tab_icon_and_colour()` (naqs-
`method`), 100 `lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`set_property()` (naqslab_devices.StaticFreqAmp`method`), 10 `set_tab_icon_and_colour()` (naqs-
`method`), 37
`set_property()` (naqs-`method`), 41
`lab_devices.TektronixTDS.labscrip_device.TDS_Scope.set_tab_icon_and_colour()` (naqs-
`method`), 105 `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`set_property()` (naqs-`method`), 44
`lab_devices.VISA.labscrip_device.VISA.set_tab_icon_and_colour()` (naqs-
`method`), 15 `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`set_sens()` (naqs-`method`), 63
`lab_devices.SR865.labscrip_device.SR865.set_tab_icon_and_colour()` (naqs-
`method`), 99 `lab_devices.SR865.blacs_tab.SR865Tab`
`set_tab_icon_and_colour()` (naqs-`method`), 97
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`set_tab_icon_and_colour()` (naqs-`method`), 102
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`set_tab_icon_and_colour()` (naqs-`method`), 79
`lab_devices.VISA.blacs_tab.VISATab`
`set_tab_icon_and_colour()` (naqs-`method`), 13
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`set_tau()` (naqslab_devices.SR865.labscrip_device.SR865`method`), 99
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`set_tab_icon_and_colour()` (naqs-`method`), 107
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409ATab`
`set_tab_icon_and_colour()` (naqs-`method`), 88
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`set_tab_icon_and_colour()` (naqs-`method`), 79
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab`
`method`), 73 `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`

```

        method), 84
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 10
        method), 86
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 10
        method), 88
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
        lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
        method), 73
        setup_string (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
        lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
        method), 68
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupply
        lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 109
        method), 18
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
        lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 80
        method), 22
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACW
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 89
        method), 26
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 90
        method), 28
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 91
        method), 31
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.PulseBlaster_No_DDS_200.blacs_worker.Pulsebla
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 74
        method), 33
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlaste
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab method), 69
        method), 51
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab
        method), 48
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor
        lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab
        method), 37
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worke
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
        method), 41
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 52
        method), 45
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10
        lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 38
        method), 63
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10
        lab_devices.SR865.blacs_tab.SR865Tab method), 42
        method), 97
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWor
        lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 46
        method), 102
        shutdown() (naqslab_devices.StaticFreqAmp method), 10
set_terminal_visible() (naqslab_devices.StaticFreqAmp method), 10
        lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWo
        lab_devices.VISA.blacs_tab.VISATab method), 65
        method), 13
        shutdown() (naqslab_devices.StaticFreqAmp method), 10

```


`lab_devices.SR865.blacs_worker.SR865Worker` `method`), 41
`method`), 99
`shutdown()` `(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker``method`), 45
`method`), 104
`shutdown()` `(naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.VISA.blacs_worker.VISAWorker` `method`), 63
`method`), 14
`shutdown_workers()` `(naqslab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab``method`), 97
`method`), 107
`shutdown_workers()` `(naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab``method`), 102
`method`), 79
`shutdown_workers()` `(naqslab_devices.VISA.blacs_tab.VISATab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab``method`), 13
`method`), 84
`shutdown_workers()` `(naqslab_devices.SignalGenerator` `(class in naqslab-`
`lab_devices.SignalGenerator.labscrip_device)`,
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` 65
`method`), 86
`shutdown_workers()` `(naqslab_devices.SignalGeneratorTab` `(class in naqslab-`
`lab_devices.SignalGenerator.blacs_tab)`,
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` 61
`method`), 88
`shutdown_workers()` `(naqslab_devices.SignalGeneratorWorker` `(class in naqslab-`
`lab_devices.SignalGenerator.blacs_worker)`,
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 73
`shutdown_workers()` `(naqslab_devices.SR865` `(class in naqslab-`
`lab_devices.SR865.labscrip_device)`, 99
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` `in naqslab-`
`method`), 68
`lab_devices.SR865.blacs_tab)`, 95
`shutdown_workers()` `(naqslab_devices.SR865Worker` `(class in naqslab-`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab``lab_devices.SR865.blacs_worker)`, 98
`method`), 18
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.KeysightDCSupply.blacs_worker.KeysightDC`
`method`), 109
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab``naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScope`
`method`), 22
`method`), 80
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab``method`), 89
`method`), 26
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B`
`method`), 90
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab``naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440A`
`method`), 28
`method`), 91
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.Pulse`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab``method`), 73
`method`), 31
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.Pu`
`method`), 74
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab``naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBl`
`method`), 33
`method`), 68
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.PulseBlasterESRPro300.blacs_worker.Pulse`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab``method`), 69
`method`), 51
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_864`
`method`), 20
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGensTab``lab_devices.SignalGenerator.BLACS.HP_8643A.HP_864`
`method`), 48
`method`), 23
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`method`), 37
`shutdown_workers()` `(naqslab_devices.start()` `(naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.K`
`method`), 52
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab``lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_S`

method), 38	(naqs-
start () (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker	start_run ())
method), 42	method), 51
start () (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker	(naqs-
method), 46	lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
start () (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker	method), 48
method), 65	start_run ()
start () (naqslab_devices.SR865.blacs_worker.SR865Worker	(naqs-
method), 99	lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker
method), 37	method), 37
start () (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker	(naqs-
method), 104	lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
start () (naqslab_devices.VISA.blacs_worker.VISAWorker	method), 41
method), 14	start_run ()
start_run ()	(naqs-
lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab)	lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 107	method), 45
start_run ()	(naqs-
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab)	lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
method), 79	method), 63
start_run ()	(naqs-
lab_devices.SR865.blacs_tab.SR865Tab	method), 97
method), 84	start_run ()
start_run ()	(naqs-
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab	method), 102
method), 86	start_run ()
start_run ()	(naqs-
lab_devices.VISA.blacs_tab.VISATab	method), 13
method), 88	state () (naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSup
start_run ()	(naqs-
property), 107	lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tight
method), 73	property), 79
start_run ()	(naqs- state () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_AC
lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker	method), 74
method), 74	state () (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTAB
start_run ()	(naqs-
property), 86	lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 68	property), 88
start_run ()	(naqs- state () (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.Pulse
lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker	method), 69
method), 69	state () (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlas
start_run ()	(naqs-
property), 68	lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
method), 18	property), 18
start_run ()	(naqs- state () (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab)	method), 22
method), 22	state () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
start_run ()	(naqs-
property), 26	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
method), 26	property), 28
start_run ()	(naqs- state () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB)	method), 28
method), 28	state () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
start_run ()	(naqs-
property), 33	lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
method), 31	property), 51
start_run ()	(naqs- state () (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.K
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB)	method), 33
method), 33	state () (naqslab_devices.SignalGenerator.BLACS.RS SMA100B.RS SMA100BWorker

179

`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`), 26
`method`), 33
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`), 26
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`), 28
`method`), 51
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`), 28
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab`), 28
`method`), 49
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`), 28
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`), 37
`method`), 37
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`), 28
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`), 41
`method`), 41
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab`), 28
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`), 49
`method`), 45
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`), 37
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`), 37
`method`), 63
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`), 41
`lab_devices.SR865.blacs_tab.SR865Tab`), 41
`method`), 97
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`), 49
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`), 45
`method`), 102
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`), 37
`lab_devices.VISA.blacs_tab.VISATab`), 63
`method`), 13
`statemachine_timeout_remove_all()` (`naqslab_devices.SR865.blacs_tab.SR865Tab`), 41
`naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`), 107
`method`), 107
`statemachine_timeout_remove_all()` (`naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`), 45
`naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`), 102
`method`), 79
`statemachine_timeout_remove_all()` (`naqslab_devices.VISA.blacs_tab.VISATab`), 63
`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409Bn467Tab`), 13
`method`), 84
`statemachine_timeout_remove_all()` (`StaticFreqAmp` (class in `naqslab_devices`), 9
`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409Bn467Tab`), 13
`naqslab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`), 107
`method`), 86
`statemachine_timeout_remove_all()` (`status_byte_labels` (class in `naqslab_devices`), 9
`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`), 16
`naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`), 102
`method`), 88
`statemachine_timeout_remove_all()` (`status_byte_labels` (class in `naqslab_devices`), 9
`naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab`), 16
`method`), 74
`statemachine_timeout_remove_all()` (`status_byte_labels` (class in `naqslab_devices`), 9
`naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`), 20
`method`), 68
`statemachine_timeout_remove_all()` (`status_byte_labels` (class in `naqslab_devices`), 9
`naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`), 18
`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`), 24
`method`), 18
`statemachine_timeout_remove_all()` (`status_byte_labels` (class in `naqslab_devices`), 9
`naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`), 28
`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`), 28
`method`), 22
`statemachine_timeout_remove_all()` (`status_byte_labels` (class in `naqslab_devices`), 9
`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`), 28
`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`), 28

`attribute)`, 31
`status_byte_labels` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`attribute)`, 33
`status_monitor()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257NTab`
`attribute)`, 51
`status_monitor()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSig`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSig40nTab`
`attribute)`, 46
`status_monitor()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`attribute)`, 35
`status_monitor()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`attribute)`, 39
`status_monitor()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`attribute)`, 45
`status_monitor()` (`naqs-`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`attribute)`, 63
`status_monitor()` (`naqs-`
`lab_devices.SR865.blacs_tab.SR865Tab`
`attribute)`, 95
`status_monitor()` (`naqs-`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`attribute)`, 100
`status_monitor()` (`naqs-`
`lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.VISA.blacs_tab.VISATab`
`attribute)`, 11
`status_widget` (`naqs-`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTa`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab`
`method)`, 107
`status_widget` (`naqs-`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`attribute)`, 79
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method)`, 74
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method)`, 68
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`attribute)`, 26
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`attribute)`, 28
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`attribute)`, 31
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`attribute)`, 33
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`attribute)`, 51
`status_widget` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`method)`, 31

`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab`
`attribute), 49` `STBui_path` `(naqs-`
`status_widget` `(naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab`
`attribute), 37` `STBui_path` `(naqs-`
`status_widget` `(naqs-` `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`attribute), 41` `STBui_path` `(naqs-`
`status_widget` `(naqs-` `lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`attribute), 96` `STBui_path` `(naqs-`
`status_widget` `(naqs-` `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`attribute), 101` `STBui_path` `(naqs-`
`status_widget` `(naqs-` `lab_devices.VISA.blacs_tab.VISATab` `at-`
`lab_devices.SR865.blacs_tab.SR865Tab` `tribute), 11`
`attribute), 97` `stop ()` `(naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseB`
`status_widget` `(naqs-` `method), 74`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `stop ()` `(naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlas`
`attribute), 102` `method), 68`
`status_widget` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.VISA.blacs_tab.VISATab` `at-` `lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTa`
`tribute), 11` `method), 108`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`attribute), 106` `method), 79`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`attribute), 77` `method), 84`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`attribute), 16` `method), 86`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`attribute), 20` `method), 88`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` `lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlas`
`attribute), 24` `method), 74`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlas`
`attribute), 26` `method), 68`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`attribute), 29` `method), 18`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`attribute), 31` `method), 22`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257MEtab` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`attribute), 49` `method), 26`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`attribute), 47` `method), 28`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BTab` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`attribute), 35` `method), 31`
`STBui_path` `(naqs-` `supports_remote_value_check ()` `(naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`

[illegible]

`to()` (`naqslab_devices.SignalGenerator.Models.E8257N` `method`), 74
`property`), 61 `terminate()` (`naqs-`
`to()` (`naqslab_devices.SignalGenerator.Models.HP_8642A` `lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlaster`
`property`), 57 `method`), 69
`to()` (`naqslab_devices.SignalGenerator.Models.HP_8642A` `terminate()` (`naqs-`
`property`), 56 `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWo`
`to()` (`naqslab_devices.SignalGenerator.Models.HP_8648A` `method`), 20
`property`), 58 `terminate()` (`naqs-`
`to()` (`naqslab_devices.SignalGenerator.Models.HP_8648B` `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWo`
`property`), 58 `method`), 23
`to()` (`naqslab_devices.SignalGenerator.Models.HP_8648C` `terminate()` (`naqs-`
`property`), 59 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Wo`
`to()` (`naqslab_devices.SignalGenerator.Models.HP_8648D` `method`), 34
`property`), 60 `terminate()` (`naqs-`
`to()` (`naqslab_devices.SignalGenerator.Models.RS_SMA100B` `lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight`
`property`), 54 `method`), 52
`to()` (`naqslab_devices.SignalGenerator.Models.RS_SMF100A` `terminate()` (`naqs-`
`property`), 53 `lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA10`
`to()` (`naqslab_devices.SignalGenerator.Models.RS_SMHU` `method`), 38
`property`), 55 `terminate()` (`naqs-`
`to()` (`naqslab_devices.SR865.labscrip_device.SR865` `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10`
`property`), 100 `method`), 42
`to()` (`naqslab_devices.StaticFreqAmp` `property`), 10 `terminate()` (`naqs-`
`to()` (`naqslab_devices.TektronixTDS.labscrip_device.TDS_Scope` `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWo`
`property`), 105 `method`), 46
`to()` (`naqslab_devices.VISA.labscrip_device.VISA` `terminate()` (`naqs-`
`property`), 15 `lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWo`
`tau` (`naqslab_devices.SR865.labscrip_device.SR865` `method`), 65
`attribute`), 99 `terminate()` (`naqs-`
`tau_changed()` (`naqs-` `lab_devices.SR865.blacs_worker.SR865Worker`
`lab_devices.SR865.blacs_tab.SR865Tab` `method`), 99
`method`), 95 `terminate()` (`naqs-`
`TDS_Scope` (`class` `in` `naqs-` `lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`lab_devices.TektronixTDS.labscrip_device`), `method`), 104
104 `terminate()` (`naqs-`
`TDS_ScopeTab` (`class` `in` `naqs-` `lab_devices.VISA.blacs_worker.VISAWorker`
`lab_devices.TektronixTDS.blacs_tab`), `method`), 14
100 `transition_to_buffered()` (`naqs-`
`TDS_ScopeWorker` (`class` `in` `naqs-` `lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTa`
`lab_devices.TektronixTDS.blacs_worker`), `method`), 108
103 `transition_to_buffered()` (`naqs-`
`terminate()` (`naqs-` `lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSuppl`
`lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWo`
`method`), 109 `transition_to_buffered()` (`naqs-`
`terminate()` (`naqs-` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWo`
`method`), 80 `method`), 79
`method`), 80 `transition_to_buffered()` (`naqs-`
`terminate()` (`naqs-` `lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWo`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTab`
`method`), 89 `transition_to_buffered()` (`naqs-`
`terminate()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWo`
`method`), 90 `method`), 84
`method`), 90 `transition_to_buffered()` (`naqs-`
`terminate()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWo`
`method`), 91 `method`), 86
`method`), 91 `transition_to_buffered()` (`naqs-`
`terminate()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlasterNoDDS200Worker`
`method`), 88

Index 185

method), 86
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab method), 53
 method), 88
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker method), 37
 method), 89
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100BWorker method), 39
 method), 89
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 41
 method), 91
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 42
 method), 74
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 43
 method), 74
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 43
 method), 68
 transition_to_manual() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 69
 transition_to_manual() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 65
 method), 18
 transition_to_manual() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 98
 method), 20
 transition_to_manual() (naqslab_devices.SR865.blacs_worker.SR865Worker method), 99
 method), 22
 transition_to_manual() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 103
 method), 24
 transition_to_manual() (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker method), 103
 method), 26
 transition_to_manual() (naqslab_devices.VISA.blacs_tab.VISATab method), 13
 method), 29
 transition_to_manual() (naqslab_devices.VISA.blacs_worker.VISAWorker method), 14
 method), 31
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Tab method), 82
 method), 33
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 76
 method), 35
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 71
 method), 35
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 71
 method), 105
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257A.Tab method), 51
 transition_to_manual() (naqslab_devices.SignalGenerator.BLACS.KeysightSigGens.KeysightSigGenTab attribute), 76
 method), 49
 transition_to_manual() (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300Worker method), 76

attribute), 71
 trigger_duration (naqs-
 lab_devices.TektronixTDS.labscript_device.TDS_Scope
 attribute), 104
 trigger_edge_type (naqs-
 lab_devices.KeysightXSeries.labscript_device.KeysightXScope
 attribute), 82
 trigger_edge_type (naqs-
 lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
 attribute), 76
 trigger_edge_type (naqs-
 lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
 attribute), 71
 trigger_edge_type (naqs-
 lab_devices.TektronixTDS.labscript_device.TDS_Scope
 attribute), 105
 trigger_minimum_duration (naqs-
 lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
 attribute), 76
 trigger_minimum_duration (naqs-
 lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
 attribute), 71
U
 update_from_settings() (naqs-
 lab_devices.KeysightDCSupply.blacs_tab.KeysightDCSupplyTab
 method), 108
 update_from_settings() (naqs-
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
 method), 79
 update_from_settings() (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
 method), 84
 update_from_settings() (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
 method), 86
 update_from_settings() (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
 method), 88
 update_from_settings() (naqs-
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
 method), 74
 update_from_settings() (naqs-
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
 method), 68
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
 method), 18
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
 method), 22
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
 method), 26
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
 method), 29
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 method), 31
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
 method), 33
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.KeysightSigGens.E8257N
 method), 51
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.KeysightSigGens.Keysight
 method), 49
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMA100B.RS_SMA100
 method), 37
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
 method), 41
 update_from_settings() (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
 method), 63
 update_from_settings() (naqs-
 lab_devices.SR865.blacs_tab.SR865Tab
 method), 98
 update_from_settings() (naqs-
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 method), 103
 update_from_settings() (naqs-
 lab_devices.VISA.blacs_tab.VISATab
 method), 13
V
 VISAWorker (class in naqs-
 lab_devices.VISA.labscript_device), 14
 VISATab (class in naqs-
 lab_devices.VISA.blacs_tab), 11
W
 wait_delay (naqs-
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
 attribute), 76
 waveform_parser() (naqs-
 lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
 method), 103
 write_both_string (naqs-
 lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupply
 attribute), 108
 write_check() (naqs-
 lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACW
 method), 89

```
write_check() (naqs-  
lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker  
method), 90  
write_check() (naqs-  
lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker  
method), 91  
write_current_string (naqs-  
lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker  
attribute), 108  
write_pb_inst_to_h5() (naqs-  
lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200  
method), 76  
write_pb_inst_to_h5() (naqs-  
lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300  
method), 71  
write_volt_string (naqs-  
lab_devices.KeysightDCSupply.blacs_worker.KeysightDCSupplyWorker  
attribute), 108
```