
naqslab_devices

Release 0.2.7

dihm

Oct 31, 2019

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	54
3.5	PulseBlaster_No_DDS_200	59
3.6	KeysightXSeries	65
3.7	NovaTechDDS	70
3.8	SR865	83
3.9	TektronixTDS	88
4	Building Documentation	95
4.1	Sphinx Environment	95
4.2	Sphinx Build	95
4.3	Adding Documentation	95
5	Indices and tables	97
6	Credits	99
	Python Module Index	101
	Index	103

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.labscrip_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_SMF100A</code>
<code>RS_SMHU</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.labscrip_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 `labscrip_device` to the correct `blacs_tab`. Note that multiple `labscrip_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': 'End of Test'}
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_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.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:\\\\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_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.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:\\\\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_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_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_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', 'b

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_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.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.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_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.labscrip_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.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_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_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_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)

```

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
**kwa

Bases: labscript_devices.PulseBlaster_No_DDS.PulseBlaster_No_DDS
description = 'SpinCore PulseBlaster ESR-PRO-300'
clock_limit = 300000000.0
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_DELAY': 0}
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: labscrip_devices.PulseBlaster_No_DDS.PulseBlaster_No_DDS_Parser
    num_flags = 24
    get_traces(add_trace, parent=None)
    labscrip_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 labscrip_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

```

Continued on next page

Table 6 – continued from previous page

```

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: labscrip_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
**
    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 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 ()

```

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_P
```

```
    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_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_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 modules covers the 409B-AC, 409B, and 440A DDS frequency synthesizers from Novatech.

Note that the 409B-AC class is largely interchangeable with the `DDS9m_labscrip_devices` class.

`naqslab_devices.NovaTechDDS.`
`blacs_tab`

`naqslab_devices.NovaTechDDS.`
`blacs_worker`

`naqslab_devices.NovaTechDDS.`
`labscrip_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_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.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_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 ()

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.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_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)

```

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 reponse 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 response 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 response 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 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_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 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_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:\\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_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 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)

```

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

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.KeysightXSeries](#), [65](#)
[naqslab_devices.KeysightXSeries.blacs_tab](#),
[65](#)
[naqslab_devices.KeysightXSeries.blacs_worker](#),
[67](#)
[naqslab_devices.KeysightXSeries.labscript_device](#),
[69](#)
[naqslab_devices.KeysightXSeries.register_classes](#),
[70](#)
[naqslab_devices.NovaTechDDS](#), [70](#)
[naqslab_devices.NovaTechDDS.blacs_tab](#),
[70](#)
[naqslab_devices.NovaTechDDS.blacs_worker](#),
[77](#)
[naqslab_devices.NovaTechDDS.labscript_device](#),
[79](#)
[naqslab_devices.NovaTechDDS.register_classes](#),
[83](#)
[naqslab_devices.NovaTechDDS.runviewer_parser](#),
[83](#)
[naqslab_devices.PulseBlaster_No_DDS_200](#),
[59](#)
[naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab](#),
[60](#)
[naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker](#),
[62](#)
[naqslab_devices.PulseBlaster_No_DDS_200.labscript_device](#),
[63](#)
[naqslab_devices.PulseBlaster_No_DDS_200.register_classes](#),
[64](#)
[naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser](#),
[64](#)
[naqslab_devices.PulseBlasterESRPro300](#),
[54](#)
[naqslab_devices.PulseBlasterESRPro300.blacs_tab](#),
[54](#)
[naqslab_devices.PulseBlasterESRPro300.blacs_worker](#),
[57](#)
[naqslab_devices.PulseBlasterESRPro300.labscript_device](#),
[58](#)
[naqslab_devices.PulseBlasterESRPro300.register_classes](#),
[59](#)
[naqslab_devices.PulseBlasterESRPro300.runviewer_parser](#),
[59](#)
[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.RS_SMF100A](#),
[35](#)
[naqslab_devices.SignalGenerator.BLACS.RS_SMHU](#),
[39](#)
[naqslab_devices.SignalGenerator.blacs_tab](#),
[49](#)
[naqslab_devices.SignalGenerator.blacs_worker](#),
[52](#)
[naqslab_devices.SignalGenerator.labscript_device](#),
[53](#)
[naqslab_devices.SignalGenerator.Models](#),
[42](#)
[naqslab_devices.SignalGenerator.register_classes](#),
[54](#)
[naqslab_devices.SR865](#), [83](#)
[naqslab_devices.SR865.blacs_tab](#), [83](#)
[naqslab_devices.SR865.blacs_worker](#), [86](#)
[naqslab_devices.SR865.labscript_device](#),
[87](#)
[naqslab_devices.SR865.register_classes](#),
[88](#)
[naqslab_devices.TektronixTDS](#), [88](#)
[naqslab_devices.TektronixTDS.blacs_tab](#),
[88](#)
[naqslab_devices.TektronixTDS.blacs_worker](#),
[91](#)
[naqslab_devices.TektronixTDS.labscript_device](#),
[92](#)
[naqslab_devices.TektronixTDS.register_classes](#),
[93](#)
[naqslab_devices.VISA](#), [10](#)
[naqslab_devices.VISA.blacs_tab](#), [11](#)
[naqslab_devices.VISA.blacs_worker](#), [13](#)
[naqslab_devices.VISA.labscript_device](#),
[14](#)
[naqslab_devices.VISA.register_classes](#),
[15](#)

INDEX

A

`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`
`method`), 23
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`method`), 65
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`method`), 24
`method`), 68
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method`), 26
`method`), 70
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`method`), 29
`method`), 73
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`method`), 31
`method`), 75
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTab`
`method`), 34
`method`), 77
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`
`method`), 35
`method`), 78
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker`
`method`), 38
`method`), 78
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 39
`method`), 60
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlasterNoDDS200Worker`
`method`), 41
`method`), 62
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 59
`method`), 55
`abort_buffered()` (naqs-
`lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 53
`method`), 57
`abort_buffered()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`method`), 84
`method`), 16
`abort_buffered()` (naqs-
`lab_devices.SR865.blacs_worker.SR865Worker`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`
`method`), 86
`method`), 19
`abort_buffered()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`method`), 89
`method`), 20
`abort_buffered()` (naqs-

`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker` method), 24
`method`), 91
`abort_buffered()` (naqslab_devices.VISA.blacs_tab.VISATab method), 11
`abort_buffered()` (naqslab_devices.VISA.blacs_worker.VISAWorker method), 14
`abort_transition_to_buffered()` (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 65
`abort_transition_to_buffered()` (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker method), 68
`abort_transition_to_buffered()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 71
`abort_transition_to_buffered()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 73
`abort_transition_to_buffered()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 75
`abort_transition_to_buffered()` (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTab method), 77
`abort_transition_to_buffered()` (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker method), 78
`abort_transition_to_buffered()` (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker method), 78
`abort_transition_to_buffered()` (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 60
`abort_transition_to_buffered()` (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker method), 62
`abort_transition_to_buffered()` (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 55
`abort_transition_to_buffered()` (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker method), 57
`abort_transition_to_buffered()` (naqslab_devices.VISA.blacs_tab.VISATab method), 16
`abort_transition_to_buffered()` (naqslab_devices.VISA.blacs_worker.VISAWorker method), 19
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker method), 20
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker method), 23
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 26
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 29
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 31
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 34
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 35
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 38
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 39
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 42
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 50
`abort_transition_to_buffered()` (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 52
`abort_transition_to_buffered()` (naqslab_devices.SR865.blacs_tab.SR865Tab method), 64
`abort_transition_to_buffered()` (naqslab_devices.SR865.blacs_worker.SR865Worker method), 66
`abort_transition_to_buffered()` (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 89
`abort_transition_to_buffered()` (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker method), 91
`acquire()` (naqslab_devices.CounterScopeChannel method), 8
`acquire()` (naqslab_devices.ScopeChannel method), 8
`acquire()` (naqslab_devices.TektronixTDS.labscript_device.TDS_ScopeWorker method), 92

```

        lab_devices.CounterScopeChannel method),
        9
        lab_devices.VISA.labscript_device.VISA
        method), 15
add_device() (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScopeTab
        method), 69
        naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
        method), 65
add_device() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_ACTab
        method), 81
        naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
        method), 71
add_device() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_TAB
        method), 80
        naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_TAB
        method), 73
add_device() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A_TAB
        method), 82
        naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440A_TAB
        method), 75
add_device() (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
        method), 63
        naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200
        method), 60
add_device() (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
        method), 58
        naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300
        method), 55
add_device() (naqslab_devices.ScopeChannel method), 8
        naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
        method), 16
add_device() (naqslab_devices.SignalGenerator.labscript_device.SignalGenerator
        method), 53
        naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
        method), 20
add_device() (naqslab_devices.SignalGenerator.Models.HP_8642A add_secondary_worker()
        method), 45
        naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
        method), 24
add_device() (naqslab_devices.SignalGenerator.Models.HP_8643A add_secondary_worker()
        method), 44
        naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
        method), 27
add_device() (naqslab_devices.SignalGenerator.Models.HP_8648A add_secondary_worker()
        method), 46
        naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
        method), 29
add_device() (naqslab_devices.SignalGenerator.Models.HP_8648B add_secondary_worker()
        method), 47
        naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
        method), 31
add_device() (naqslab_devices.SignalGenerator.Models.HP_8648C add_secondary_worker()
        method), 48
        naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
        method), 35
add_device() (naqslab_devices.SignalGenerator.Models.HP_8648D add_secondary_worker()
        method), 49
        naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
        method), 39
add_device() (naqslab_devices.SignalGenerator.Models.RS_SMF100A add_secondary_worker()
        method), 43
        naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
        method), 50
add_device() (naqslab_devices.SignalGenerator.Models.RS_SMHU add_secondary_worker()
        method), 43
        naqslab_devices.SR865.blacs_tab.SR865Tab
        method), 84
add_device() (naqslab_devices.SR865.labscript_device.SR865 add_secondary_worker()
        method), 87
        naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
        method), 89
add_device() (naqslab_devices.StaticFreqAmp method), 10
        naqslab_devices.VISA.blacs_tab.VISATab
        method), 11
add_device() (naqslab_devices.TektronixTDS.labscript_device.TDS_Scope method), 92
        allowed_children
        naqslab_devices.CounterScopeChannel
        at-

```

	tribute), 9	lab_devices.VISA.labscript_device.VISA
allowed_children	(naqs- lab_devices.KeysightXSeries.labscript_device.KeysightXScope attribute), 69	attribute), 15
allowed_children	(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409Bts attribute), 81	lab_devices.SignalGenerator.labscript_device.SignalGenerator attribute), 53
allowed_children	(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409Bts attribute), 80	lab_devices.SignalGenerator.Models.HP_8642A attribute), 45
allowed_children	(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409Bts attribute), 82	lab_devices.SignalGenerator.Models.HP_8643A attribute), 44
allowed_children	(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409Bts attribute), 82	lab_devices.SignalGenerator.Models.HP_8648A attribute), 46
allowed_children	(naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200 attribute), 63	lab_devices.SignalGenerator.Models.HP_8648B attribute), 47
allowed_children	(naqs- lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300 attribute), 58	lab_devices.SignalGenerator.Models.HP_8648C attribute), 48
allowed_children	(naqs- lab_devices.ScopeChannel attribute), 8	amp_limits (naqs- lab_devices.SignalGenerator.Models.HP_8648D attribute), 48
allowed_children	(naqs- lab_devices.SignalGenerator.labscript_device.SignalGenerator attribute), 53	amp_limits (naqs- lab_devices.SignalGenerator.Models.RS_SMF100A attribute), 43
allowed_children	(naqs- lab_devices.SignalGenerator.Models.HP_8642A attribute), 45	amp_limits (naqs- lab_devices.SignalGenerator.Models.RS_SMHU attribute), 43
allowed_children	(naqs- lab_devices.SignalGenerator.Models.HP_8643A attribute), 44	amp_parser() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor method), 19
allowed_children	(naqs- lab_devices.SignalGenerator.Models.HP_8648A attribute), 46	amp_parser() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor method), 23
allowed_children	(naqs- lab_devices.SignalGenerator.Models.HP_8648B attribute), 47	amp_parser() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worke method), 34
allowed_children	(naqs- lab_devices.SignalGenerator.Models.HP_8648C attribute), 48	amp_parser() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10 method), 37
allowed_children	(naqs- lab_devices.SignalGenerator.Models.HP_8648D attribute), 49	amp_parser() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWor method), 41
allowed_children	(naqs- lab_devices.SignalGenerator.Models.RS_SMF100A attribute), 43	amp_parser() (naqs- lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWo method), 52
allowed_children	(naqs- lab_devices.SR865.labscript_device.SR865 attribute), 87	amp_query_string (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWor attribute), 19
allowed_children	(naqs- lab_devices.StaticFreqAmp attribute), 10	amp_query_string (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWor attribute), 23
allowed_children	(naqs- lab_devices.TektronixTDS.labscript_device.TDS_Scope attribute), 92	amp_query_string (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worke attribute), 34
allowed_children	(naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10 attribute), 37	amp_query_string (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10 attribute), 37

107

`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 84
`method`), 50 `auto_place_widgets()` (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
`lab_devices.SR865.blacs_tab.SR865Tab` method), 89
`method`), 84 `auto_place_widgets()` (naqslab_devices.VISA.blacs_tab.VISATab
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 11
`method`), 89
`auto_create_widgets()` (naqslab_devices.VISA.blacs_tab.VISATab
`method`), 11
`auto_place_widgets()` (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
`method`), 65
`auto_place_widgets()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
`method`), 71
`auto_place_widgets()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
`method`), 73
`auto_place_widgets()` (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
`method`), 75
`auto_place_widgets()` (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
`method`), 60
`auto_place_widgets()` (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
`method`), 55
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
`method`), 17
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
`method`), 20
`auto_place_widgets()` (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
`method`), 49
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
`method`), 24
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
`method`), 27
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
`method`), 29
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
`method`), 31
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
`method`), 35
`auto_place_widgets()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
`method`), 39
`auto_place_widgets()` (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
`method`), 50
`auto_place_widgets()` (naqslab_devices.SR865.blacs_tab.SR865Tab
`method`), 20

`base_min(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`attribute), 24`
`base_min(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`attribute), 26`
`base_min(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`attribute), 29`
`base_min(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`attribute), 31`
`base_min(naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`attribute), 35`
`base_min(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`attribute), 39`
`base_min(naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`attribute), 49`
`base_step(naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`attribute), 16`
`base_step(naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`attribute), 20`
`base_step(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`attribute), 24`
`base_step(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`attribute), 27`
`base_step(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`attribute), 29`
`base_step(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`attribute), 31`
`base_step(naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`attribute), 35`
`base_step(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`attribute), 39`
`base_step(naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`attribute), 49`
`base_units`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`attribute), 16`
`base_units`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`attribute), 20`
`base_units`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`attribute), 24`
`base_units`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`attribute), 27`
`base_units`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`attribute), 29`
`base_units`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`attribute), 31`
`base_units`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`attribute), 35`
`base_units`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`attribute), 39`
`base_units`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`attribute), 49`

```

        lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker, 68
        method), 20
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster
        lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker, 62
        method), 23
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlaster
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648AWorker, 57
        method), 24
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker, 19
        method), 27
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CWorker, 23
        method), 29
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DWorker, 34
        method), 31
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
        lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker, 37
        method), 34
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker, 35
        method), 35
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
        lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker, 38
        method), 38
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.SR865.blacs_worker.SR865Worker
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker, 86
        method), 39
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
        lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker, 91
        method), 42
        check_status() (naqs-
check_remote_values() (naqs- lab_devices.VISA.blacs_worker.VISAWorker
        lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab, 13
        method), 50
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
        lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker, 65
        method), 52
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
        lab_devices.SR865.blacs_tab.SR865Tab, 71
        method), 84
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
        lab_devices.SR865.blacs_worker.SR865Worker, 73
        method), 86
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
        lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab, 75
        method), 89
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
        lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker, 60
        method), 91
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
        lab_devices.VISA.blacs_tab.VISATab, 55
        method), 11
        check_time() (naqs-
check_remote_values() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
        lab_devices.VISA.blacs_worker.VISAWorker, 17
        method), 13
        check_time() (naqs-
check_status() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
        lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker, 20

```

`check_time()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 27
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 29
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648GTab` method), 31
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` method), 35
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` method), 35
`clean_ui_on_restart()` (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` method), 50
`clean_ui_on_restart()` (naqslab_devices.SR865.blacs_tab.SR865Tab
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 84
`clean_ui_on_restart()` (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
`lab_devices.SR865.blacs_tab.SR865Tab` method), 89
`clean_ui_on_restart()` (naqslab_devices.VISA.blacs_tab.VISATab
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 11
`clear()` (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeTab
`lab_devices.VISA.blacs_tab.VISATab` method), 68
`clear()` (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642A
`lab_devices.VISA.blacs_tab.VISATab` method), 19
`clear()` (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` method), 23
`clear()` (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B.ACTab` method), 34
`clear()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 38
`clear()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 42
`clear()` (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorTab
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 52
`clear()` (naqslab_devices.SR865.blacs_worker.SR865Worker
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 75
`clear()` (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` method), 60
`clear()` (naqslab_devices.VISA.blacs_worker.VISAWorker
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` method), 14
`lab_devices.NovaTechDDS.labscript_device.NovaTech409B
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 55
lab_devices.NovaTechDDS.labscript_device.NovaTech409B.AC
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 81
lab_devices.NovaTechDDS.labscript_device.NovaTech409B.AC
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 80
lab_devices.NovaTechDDS.labscript_device.NovaTech440A
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 20
lab_devices.NovaTechDDS.labscript_device.NovaTech440A
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 82
lab_devices.NovaTechDDS.labscript_device.NovaTech409B
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 24
lab_devices.NovaTechDDS.labscript_device.NovaTech409B
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab attribute), 81`

```

clock_limit (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_ACTab), 39
attribute), 80 close_tab() (naqs-
clock_limit (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
lab_devices.NovaTechDDS.labscrip_device.NovaTech440ATab), 50
attribute), 82 close_tab() (naqs-
clock_limit (naqs- lab_devices.SR865.blacs_tab.SR865Tab
lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200
attribute), 63 close_tab() (naqs-
clock_limit (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300
attribute), 58 close_tab() (naqs-
clock_resolution (naqs- lab_devices.VISA.blacs_tab.VISATab
lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200
attribute), 63 coerce_sens() (naqs-
clock_resolution (naqs- lab_devices.SR865.blacs_worker.SR865Worker
lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300
attribute), 58 coerce_tau() (naqs-
close_tab() (naqs- lab_devices.SR865.blacs_worker.SR865Worker
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab), 86
method), 65 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab), 66
method), 71 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab), 71
method), 73 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab), 73
method), 75 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 60 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 55 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab), 55
method), 17 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab), 17
method), 21 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 21
method), 24 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 24
method), 27 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 27
method), 29 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 29
method), 32 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
method), 35 connect_restart_receiver() (naqs-
close_tab() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10

```


[illegible]

`lab_devices.PulseBlasterESRPro300.blacs_tab.VISATab`
`attribute), 58`
`count()` (`naqslab_devices.CounterScopeChannel` `create_analog_widgets()` (`naqs-`
`method), 8` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`CounterScopeChannel` (`class` `in` `naqs-` `method), 66`
`lab_devices), 8` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` `method), 71`
`method), 66` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` `method), 73`
`method), 71` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` `method), 75`
`method), 73` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` `method), 60`
`method), 75` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method), 60` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method), 55` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` `method), 21`
`method), 17` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` `method), 25`
`method), 21` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` `method), 27`
`method), 25` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` `method), 29`
`method), 27` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` `method), 32`
`method), 29` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` `method), 36`
`method), 32` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method), 36` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` `method), 50`
`method), 39` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` `method), 84`
`method), 50` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.SR865.blacs_tab.SR865Tab` `method), 89`
`method), 84` `create_analog_widgets()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `method), 11`
`method), 89` `create_dds_outputs()` (`naqs-`
`create_analog_outputs()` (`naqs-` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`

Index	115
--------------	------------

`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 61
`method`), 75 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 60 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 55 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` method), 21
`method`), 17 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` method), 25
`method`), 21 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 27
`method`), 25 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 29
`method`), 27 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` method), 32
`method`), 29 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` method), 36
`method`), 32 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` method), 36
`method`), 36 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` method), 50
`method`), 39 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 84
`method`), 50 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.SR865.blacs_tab.SR865Tab` method), 89
`method`), 84 `create_digital_outputs()` (naqs-
`create_device_properties()` (naqs- `lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 12
`method`), 89 `create_digital_widgets()` (naqs-
`create_device_properties()` (naqs- `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`lab_devices.VISA.blacs_tab.VISATab` method), 66
`method`), 12 `create_digital_widgets()` (naqs-
`create_digital_outputs()` (naqs- `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` method), 71
`method`), 66 `create_digital_widgets()` (naqs-
`create_digital_outputs()` (naqs- `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` method), 73
`method`), 71 `create_digital_widgets()` (naqs-
`create_digital_outputs()` (naqs- `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 75
`method`), 73 `create_digital_widgets()` (naqs-
`create_digital_outputs()` (naqs- `lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 61
`method`), 75 `create_digital_widgets()` (naqs-
`create_digital_outputs()` (naqs- `lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`

117

```

        method), 25
create_image_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
        method), 30
        method), 27
create_image_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
        method), 32
        method), 29
create_image_widgets() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
        method), 36
        method), 32
create_image_widgets() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
        method), 36
        method), 36
create_image_widgets() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
        method), 50
        method), 40
create_image_widgets() (naqslab_devices.SR865.blacs_tab.SR865Tab
        method), 84
        method), 50
create_image_widgets() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
        method), 89
        method), 84
create_image_widgets() (naqslab_devices.VISA.blacs_tab.VISATab
        method), 12
        method), 89
create_image_widgets() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
        method), 66
        method), 12
create_property_widgets() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
        method), 71
        method), 66
create_property_widgets() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
        method), 73
        method), 71
create_property_widgets() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
        method), 75
        method), 73
create_property_widgets() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
        method), 61
        method), 75
create_property_widgets() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
        method), 61
        method), 61
create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
        method), 55
        method), 55
create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
        method), 21
        method), 17
create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
        method), 25
        method), 21
create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
        method), 27
        method), 25
create_property_widgets() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
        method), 30
        method), 27
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
        create_worker() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
        create_worker() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
        create_worker() (naqslab_devices.SR865.blacs_tab.SR865Tab
        create_worker() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
        create_worker() (naqslab_devices.VISA.blacs_tab.VISATab
        create_worker() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
        create_worker() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
        create_worker() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
        create_worker() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
        create_worker() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
        create_worker() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
        create_worker() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab

```

`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`method), 32`

`create_worker()` (naqs- description (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method), 36`

`create_worker()` (naqs- description (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method), 40`

`create_worker()` (naqs- description (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method), 50`

`create_worker()` (naqs- description (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`method), 84`

`create_worker()` (naqs- description (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method), 89`

`create_worker()` (naqs- description (naqs-
`lab_devices.VISA.blacs_tab.VISATab`
`method), 12`

D

`description` (naqs- description (naqs-
`lab_devices.CounterScopeChannel`
`tribute), 8`

`description` (naqs- device_name() (naqs-
`lab_devices.KeysightXSeries.labscrip_device.KeysightXScopeTab`
`attribute), 69`

`description` (naqs- device_name() (naqs-
`lab_devices.NovaTechDDS.labscrip_device.NovaTech409B`
`attribute), 81`

`description` (naqs- device_name() (naqs-
`lab_devices.NovaTechDDS.labscrip_device.NovaTech409B`
`attribute), 79`

`description` (naqs- device_name() (naqs-
`lab_devices.NovaTechDDS.labscrip_device.NovaTech440A`
`attribute), 82`

`description` (naqs- device_name() (naqs-
`lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200`
`attribute), 63`

`description` (naqs- device_name() (naqs-
`lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300`
`attribute), 58`

`description` (naqslab_devices.ScopeChannel at- device_name() (naqs-
`tribute), 8`

`description` (naqs- property), 17
`lab_devices.SignalGenerator.labscrip_device.SignalGenerator`
`attribute), 53`

`description` (naqs- property), 21
`lab_devices.SignalGenerator.Models.HP_8642A`
`attribute), 45`

`description` (naqs- property), 25
`lab_devices.SignalGenerator.Models.HP_8643A`
`attribute), 44`

`description` (naqs- property), 27
`lab_devices.SignalGenerator.Models.HP_8648A`
`attribute), 46`

`description` (naqs- property), 30

[device_name\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8648A.HP_8648ATab](#) [method](#)), 21
[lab_devices.SignalGenerator.BLACS.HP_8648A.HP_8648ATab](#) [property](#)), 32
[device_name\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab](#) [method](#)), 25
[lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab](#) [property](#)), 36
[device_name\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab](#) [method](#)), 27
[lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab](#) [property](#)), 40
[device_name\(\)](#) ([naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab](#) [method](#)), 30
[lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab](#) [property](#)), 50
[device_name\(\)](#) ([naqslab_devices.SR865.blacs_tab.SR865Tab](#) [method](#)), 32
[lab_devices.SR865.blacs_tab.SR865Tab](#) [disconnect_restart_receiver\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab](#) [property](#)), 84
[device_name\(\)](#) ([naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab](#) [method](#)), 36
[lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab](#) [connect_restart_receiver\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab](#) [property](#)), 89
[device_name\(\)](#) ([naqslab_devices.VISA.blacs_tab.VISATab](#) [property](#)), 12
[lab_devices.VISA.blacs_tab.VISATab](#) [disconnect_restart_receiver\(\)](#) ([naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab](#) [property](#)), 50
[dig_command\(\)](#) ([naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker](#) [method](#)), 50
[lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker](#) [disconnect_restart_receiver\(\)](#) ([naqslab_devices.SR865.blacs_tab.SR865Tab](#) [property](#)), 68
[digital_pod_parser\(\)](#) ([naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker](#) [method](#)), 84
[lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker](#) [disconnect_restart_receiver\(\)](#) ([naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab](#) [property](#)), 68
[direct_outputs\(\)](#) ([naqslab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200](#) [method](#)), 89
[lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200](#) [property](#)), 63
[direct_outputs\(\)](#) ([naqslab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300](#) [method](#)), 12
[lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300](#) [property](#)), 58
[disable\(\)](#) ([naqslab_devices.StaticFreqAmp](#) [method](#)), 69
[lab_devices.StaticFreqAmp](#) [do_checks\(\)](#) ([naqslab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200](#) [method](#)), 10
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab](#) [method](#)), 63
[lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab](#) [do_checks\(\)](#) ([naqslab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300](#) [method](#)), 66
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab](#) [method](#)), 58
[lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab](#) [do_checks\(\)](#) ([naqslab_devices.TektronixTDS.labscrip_device.TDS_ScopeTab](#) [method](#)), 71
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab](#) [method](#)), 92
[lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab](#) [method](#)), 73
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.NovaTech440ATab](#) [method](#)), 75
[lab_devices.NovaTech440ATab](#) [enable\(\)](#) ([naqslab_devices.StaticFreqAmp](#) [method](#)), 10
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab](#) [method](#)), 61
[lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab](#) [error_message\(\)](#) ([naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab](#) [property](#)), 66
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab](#) [method](#)), 55
[lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab](#) [error_message\(\)](#) ([naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab](#) [property](#)), 71
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab](#) [method](#)), 17
[lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab](#) [error_message\(\)](#) ([naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab](#) [property](#)), 73
[disconnect_restart_receiver\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab](#) [method](#)), 17
[lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab](#) [error_message\(\)](#) ([naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab](#) [property](#)), 73

property), 75
 error_message() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab
 property), 61
 error_message() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
 property), 55
 error_message() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab), 61
 property), 17
 error_message() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab), 55
 property), 21
 error_message() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 17
 property), 25
 error_message() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 21
 property), 27
 error_message() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 25
 property), 30
 error_message() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 27
 property), 32
 error_message() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
 property), 36
 error_message() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 32
 property), 40
 error_message() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10
 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
 property), 50
 error_message() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
 lab_devices.SR865.blacs_tab.SR865Tab
 property), 84
 error_message() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 property), 89
 error_message() (naqslab_devices.SR865.blacs_tab.SR865Tab
 lab_devices.VISA.blacs_tab.VISATab
 property), 12
 error_parser() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
 method), 90
 esr_mask(naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
 attribute), 67
 ESRPro(naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker
 attribute), 57
 F
 finalise_close_tab() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
 method), 66
 finalise_close_tab() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
 method), 71
 finalise_restart() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
 method), 71

`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` property), 66
`method`), 76
`force_full_buffered_reprogram()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 61
`force_full_buffered_reprogram()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 55
`force_full_buffered_reprogram()` (naqs-
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` property), 76
`method`), 17
`force_full_buffered_reprogram()` (naqs-
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` property), 61
`method`), 21
`force_full_buffered_reprogram()` (naqs-
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300_Tab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` property), 55
`method`), 25
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` property), 17
`method`), 27
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` property), 21
`method`), 30
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` property), 25
`method`), 32
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` property), 36
`method`), 36
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` property), 30
`method`), 40
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` property), 32
`method`), 50
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`lab_devices.SR865.blacs_tab.SR865Tab` property), 36
`method`), 85
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` property), 40
`method`), 90
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.VISA.blacs_tab.VISATab` property), 50
`method`), 12
`force_full_buffered_reprogram()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200`
`method`), 63
`force_full_buffered_reprogram()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300`
`method`), 58
`force_full_buffered_reprogram()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab` prop-
`lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200`
`method`), 63
`freq_limits` (naqs-
`lab_devices.SignalGenerator.labscript_device.SignalGenerator`
`lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300`
`method`), 58
`freq_limits` (naqs-
`lab_devices.SignalGenerator.Models.HP_8642A`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` attribute), 45

freq_limits (naqslab_devices.SignalGenerator.Models.HP_8643A attribute), 19
 freq_limits (naqslab_devices.SignalGenerator.Models.HP_8648A attribute), 22
 freq_limits (naqslab_devices.SignalGenerator.Models.HP_8648B attribute), 34
 freq_limits (naqslab_devices.SignalGenerator.Models.HP_8648C attribute), 37
 freq_limits (naqslab_devices.SignalGenerator.Models.HP_8648D attribute), 41
 freq_limits (naqslab_devices.SignalGenerator.Models.RS_SMF100A attribute), 52
 freq_limits (naqslab_devices.SignalGenerator.Models.RS_SMHU attribute), 43
 freq_parser (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker method), 19
 freq_parser (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker method), 23
 freq_parser (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 34
 freq_parser (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 37
 freq_parser (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 41
 freq_parser (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 52
 freq_query_string (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker attribute), 19
 freq_query_string (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker attribute), 23
 freq_query_string (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker attribute), 34
 freq_query_string (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker attribute), 37
 freq_query_string (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker attribute), 41
 freq_query_string (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker attribute), 52
 freq_write_string (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker attribute), 19
 freq_write_string (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker attribute), 22
 freq_write_string (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker attribute), 34
 freq_write_string (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker attribute), 37
 freq_write_string (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker attribute), 41
 freq_write_string (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker attribute), 52
 generate_code (naqslab_devices.CounterScopeChannel method), 9
 generate_code (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope method), 69
 generate_code (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B method), 81
 generate_code (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC method), 80
 generate_code (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A method), 82
 generate_code (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster method), 63
 generate_code (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlaster method), 58
 generate_code (naqslab_devices.ScopeChannel method), 8
 generate_code (naqslab_devices.SignalGenerator.labscript_device.SignalGenerator method), 53
 generate_code (naqslab_devices.SignalGenerator.Models.HP_8642A attribute), 45
 generate_code (naqslab_devices.SignalGenerator.Models.HP_8643A attribute), 44
 generate_code (naqslab_devices.SignalGenerator.Models.HP_8648A attribute), 46
 generate_code (naqslab_devices.SignalGenerator.Models.HP_8648B attribute), 47

<code>lab_devices.SignalGenerator.Models.HP_8648C</code>	<code>lab_devices.SignalGenerator.Models.HP_8643A</code>
<code>method), 48</code>	<code>method), 44</code>
<code>generate_code()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.SignalGenerator.Models.HP_8648D</code>	<code>lab_devices.SignalGenerator.Models.HP_8648A</code>
<code>method), 49</code>	<code>method), 46</code>
<code>generate_code()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.SignalGenerator.Models.RS_SMF100A</code>	<code>lab_devices.SignalGenerator.Models.HP_8648B</code>
<code>method), 43</code>	<code>method), 47</code>
<code>generate_code()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.SignalGenerator.Models.RS_SMHU</code>	<code>lab_devices.SignalGenerator.Models.HP_8648C</code>
<code>method), 44</code>	<code>method), 48</code>
<code>generate_code()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.SR865.labscript_device.SR865</code>	<code>lab_devices.SignalGenerator.Models.HP_8648D</code>
<code>method), 87</code>	<code>method), 49</code>
<code>generate_code()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.StaticFreqAmp method), 10</code>	<code>lab_devices.SignalGenerator.Models.RS_SMF100A</code>
<code>generate_code()</code>	<code>method), 43</code>
<code>lab_devices.TektronixTDS.labscript_device.TDS_Scope</code>	<code>get_all_children()</code>
<code>method), 92</code>	<code>(naqs- lab_devices.SignalGenerator.Models.RS_SMHU</code>
<code>generate_code()</code>	<code>method), 44</code>
<code>lab_devices.VISA.labscript_device.VISA</code>	<code>get_all_children()</code>
<code>method), 15</code>	<code>(naqs- lab_devices.SR865.labscript_device.SR865</code>
<code>generate_registers()</code>	<code>method), 88</code>
<code>lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200</code>	<code>(naqs- lab_devices.StaticFreqAmp method), 10</code>
<code>method), 63</code>	<code>lab_devices.StaticFreqAmp method), 10</code>
<code>generate_registers()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300</code>	<code>(naqs- lab_devices.TektronixTDS.labscript_device.TDS_Scope</code>
<code>method), 58</code>	<code>method), 92</code>
<code>get_all_children()</code>	<code>(naqs- get_all_children()</code>
<code>lab_devices.CounterScopeChannel method), 9</code>	<code>lab_devices.VISA.labscript_device.VISA</code>
<code>9</code>	<code>method), 15</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.KeysightXSeries.labscript_device.KeysightXScope</code>	<code>(naqs- lab_devices.CounterScopeChannel method), 9</code>
<code>method), 69</code>	<code>9</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.NovaTechDDS.labscript_device.NovaTech409Bb</code>	<code>(naqs- lab_devices.KeysightXSeries.labscript_device.KeysightXScope</code>
<code>method), 81</code>	<code>method), 69</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.NovaTechDDS.labscript_device.NovaTech409Bb</code>	<code>(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B</code>
<code>method), 80</code>	<code>method), 81</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.NovaTechDDS.labscript_device.NovaTech440A</code>	<code>(naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B</code>
<code>method), 82</code>	<code>method), 80</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200</code>	<code>(naqs- lab_devices.NovaTech440A</code>
<code>method), 63</code>	<code>method), 82</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300</code>	<code>(naqs- lab_devices.PulseBlaster_No_DDS_200.labscript_device.Pulse</code>
<code>method), 58</code>	<code>method), 64</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.ScopeChannel method), 8</code>	<code>lab_devices.PulseBlasterESRPro300.labscript_device.Pulse</code>
<code>get_all_children()</code>	<code>method), 58</code>
<code>lab_devices.SignalGenerator.labscript_device.SignalGenerator</code>	<code>get_all_outputs()</code>
<code>method), 53</code>	<code>(naqs- lab_devices.ScopeChannel method), 8</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
<code>lab_devices.SignalGenerator.Models.HP_8642A</code>	<code>(naqs- lab_devices.SignalGenerator.labscript_device.SignalGenerator</code>
<code>method), 45</code>	<code>method), 53</code>
<code>get_all_children()</code>	<code>(naqs- get_all_outputs()</code>
	<code>(naqs-</code>

`lab_devices.SignalGenerator.Models.HP_8642A.get_all_save_data()` (naqs-
`method`), 45 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`get_all_outputs()` (naqs-`method`), 25
`lab_devices.SignalGenerator.Models.HP_8643A.get_all_save_data()` (naqs-
`method`), 44 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`get_all_outputs()` (naqs-`method`), 27
`lab_devices.SignalGenerator.Models.HP_8648A.get_all_save_data()` (naqs-
`method`), 46 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`get_all_outputs()` (naqs-`method`), 30
`lab_devices.SignalGenerator.Models.HP_8648B.get_all_save_data()` (naqs-
`method`), 47 `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`get_all_outputs()` (naqs-`method`), 32
`lab_devices.SignalGenerator.Models.HP_8648C.get_all_save_data()` (naqs-
`method`), 48 `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`get_all_outputs()` (naqs-`method`), 36
`lab_devices.SignalGenerator.Models.HP_8648D.get_all_save_data()` (naqs-
`method`), 49 `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`get_all_outputs()` (naqs-`method`), 40
`lab_devices.SignalGenerator.Models.RS_SMF100A.get_all_save_data()` (naqs-
`method`), 43 `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`get_all_outputs()` (naqs-`method`), 50
`lab_devices.SignalGenerator.Models.RS_SMHU.get_all_save_data()` (naqs-
`method`), 44 `lab_devices.SR865.blacs_tab.SR865Tab`
`get_all_outputs()` (naqs-`method`), 85
`lab_devices.SR865.labscript_device.SR865.get_all_save_data()` (naqs-
`method`), 88 `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`get_all_outputs()` (naqs-`method`), 90
`lab_devices.StaticFreqAmp.method`), 10 `get_all_save_data()` (naqs-
`get_all_outputs()` (naqs-`lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.TektronixTDS.labscript_device.TDS_Scope.method`), 12
`method`), 92 `get_builtin_save_data()` (naqs-
`get_all_outputs()` (naqs-`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`lab_devices.VISA.labscript_device.VISA.method`), 66
`method`), 15 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab.method`), 71
`method`), 66 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab.method`), 73
`method`), 71 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab.method`), 76
`method`), 73 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab.method`), 61
`method`), 76 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab`
`method`), 61 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 55 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab.method`), 21
`method`), 17 `get_builtin_save_data()` (naqs-
`get_all_save_data()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab.method`), 25
`method`), 21 `get_builtin_save_data()` (naqs-

```

lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 30
method), 27 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648GTab method), 32
method), 30 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 36
method), 32 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
method), 36 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 51
method), 40 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.SR865.blacs_tab.SR865Tab
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 85
method), 51 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.SR865.blacs_tab.SR865Tab method), 90
method), 85 get_channel () (naqs-
get_builtin_save_data () (naqs- lab_devices.VISA.blacs_tab.VISATab
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 12
method), 90 get_child_from_connection_table ()
get_builtin_save_data () (naqs- (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTa
lab_devices.VISA.blacs_tab.VISATab method), 66
method), 12 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_AC
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 72
method), 66 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 74
method), 71 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 76
method), 74 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseB
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 61
method), 76 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlas
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 61 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_864
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 56 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_864
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 21
method), 17 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 25
method), 21 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 28
method), 25 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 30
method), 28 get_child_from_connection_table ()
get_channel () (naqs- (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648GTab method), 32

```

```

get_child_from_connection_table()      lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
(naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
method), 36                          get_front_panel_properties()      (naqs-
get_child_from_connection_table()      lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 40                          get_front_panel_properties()      (naqs-
get_child_from_connection_table()      lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
(naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
method), 51                          get_front_panel_properties()      (naqs-
get_child_from_connection_table()      lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
(naqslab_devices.SR865.blacs_tab.SR865Tab
method), 85                          get_front_panel_properties()      (naqs-
get_child_from_connection_table()      lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
(naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTabmethod), 30
method), 90                          get_front_panel_properties()      (naqs-
get_child_from_connection_table()      lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
(naqslab_devices.VISA.blacs_tab.VISATab
method), 12                          get_front_panel_properties()      (naqs-
get_default_unit_conversion_classes()  lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
(naqslab_devices.NovaTechDDS.labscrip_device.NovaTech409B_ACTab
method), 81                          get_front_panel_properties()      (naqs-
get_default_unit_conversion_classes()  lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
(naqslab_devices.NovaTechDDS.labscrip_device.NovaTech409B_ACTab
method), 80                          get_front_panel_properties()      (naqs-
get_default_unit_conversion_classes()  lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
(naqslab_devices.NovaTechDDS.labscrip_device.NovaTech440ATab
method), 82                          get_front_panel_properties()      (naqs-
get_direct_outputs()                  (naqs- lab_devices.SR865.blacs_tab.SR865Tab
lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200
method), 64                          get_front_panel_properties()      (naqs-
get_direct_outputs()                  (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300
method), 58                          get_front_panel_properties()      (naqs-
get_flag_number()                     (naqs- lab_devices.VISA.blacs_tab.VISATab
lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200
method), 64                          get_front_panel_values()          (naqs-
get_flag_number()                     (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300
method), 58                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTabmethod), 72
method), 66                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTabmethod), 74
method), 72                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 76
method), 74                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 61
method), 76                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 61                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 56                          get_front_panel_values()          (naqs-
get_front_panel_properties()           (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab

```


get_property() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A method), 47
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8648C method), 82
 get_property() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab method), 48
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 61
 get_property() (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200 method), 49
 get_property() (naqslab_devices.SignalGenerator.Models.RS_SMF100A method), 64
 get_property() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab method), 43
 get_property() (naqslab_devices.SignalGenerator.Models.RS_SMHU method), 56
 get_property() (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300 method), 44
 get_property() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 58
 get_property() (naqslab_devices.ScopeChannel method), 85
 get_property() (naqslab_devices.SR865.labscript_device.SR865 method), 8
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 88
 get_property() (naqslab_devices.StaticFreqAmp method), 18
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 10
 get_property() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 21
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 90
 get_property() (naqslab_devices.TektronixTDS.labscript_device.TDS_Scope method), 25
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 93
 get_property() (naqslab_devices.VISA.blacs_tab.VISATab method), 28
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 12
 get_property() (naqslab_devices.VISA.labscript_device.VISA method), 30
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 15
 get_property() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 32
 get_property() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 66
 get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 36
 get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 72
 get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 40
 get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 74
 get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 51
 get_property() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 76
 get_property() (naqslab_devices.SignalGenerator.labscript_device.SignalGenerator method), 53
 get_property() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster method), 61
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8642A.HP_8642ATab method), 45
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8643A.HP_8643ATab method), 56
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8643A.HP_8643ATab method), 45
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8642A.HP_8642ATab method), 18
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8648A.HP_8648ATab method), 46
 get_property() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 21
 get_property() (naqslab_devices.SignalGenerator.Models.HP_8648B.HP_8648BTab method), 21

`lab_devices.SignalGenerator.BLACS.HP_8648A.Tab` method), 28
`get_tab_layout()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648CTab` method), 25
`get_save_data()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648B.Tab` method), 30
`get_tab_layout()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648D.Tab` method), 28
`get_save_data()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648G.Tab` method), 32
`get_tab_layout()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648H.Tab` method), 36
`get_save_data()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A.Tab` method), 36
`get_tab_layout()` (naqs-`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU.Tab` method), 51
`get_save_data()` (naqs-`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 85
`get_tab_layout()` (naqs-`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 90
`get_save_data()` (naqs-`lab_devices.SR865.blacs_tab.SR865Tab` method), 85
`get_tab_layout()` (naqs-`lab_devices.VISA.blacs_tab.VISATab` method), 12
`get_traces()` (naqs-`lab_devices.NovaTechDDS.runviewer_parser.NovaTech409B_ACTab` method), 83
`get_tab_layout()` (naqs-`lab_devices.NovaTechDDS.runviewer_parser.NovaTech409BPa` method), 83
`get_traces()` (naqs-`lab_devices.NovaTechDDS.runviewer_parser.NovaTech440APa` method), 83
`get_traces()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.runviewer_parser.Puls` method), 64
`get_traces()` (naqs-`lab_devices.PulseBlasterESRPro300.runviewer_parser.PulseBl` method), 59
`get_tab_layout()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab` method), 61
`get_tab_layout()` (naqs-`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` method), 66
`get_tab_layout()` (naqs-`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` method), 56
`get_tab_layout()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab` method), 72
`hide_error()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` method), 18
`hide_error()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` method), 74
`hide_error()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` method), 21
`hide_error()` (naqs-`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` method), 76
`hide_error()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 25
`hide_error()` (naqs-`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster` method), 61
`hide_error()` (naqs-`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 30

`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method), 56` `HP_8648ATab` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648`),
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`method), 18` `HP_8648B` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.Models`),
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`method), 21` `HP_8648BTab` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648`),
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`method), 25` `HP_8648C` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.Models`),
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`method), 28` `HP_8648CTab` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648`),
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`method), 30` `HP_8648D` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.Models`),
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`method), 32` `HP_8648DTab` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648`),
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method), 36` `HP_8648Worker` (class in `naqs-`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.BLACS.HP_8648`),
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method), 40`
`hide_error()` (`naqs-` `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` (`naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScope`
`method), 51` `attribute), 65`
`hide_error()` (`naqs-` `ICON_BUSY` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B`
`lab_devices.SR865.blacs_tab.SR865Tab` `attribute), 70`
`method), 85` `ICON_BUSY` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B`
`hide_error()` (`naqs-` `attribute), 72`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `ICON_BUSY` (`naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440AT`
`method), 90` `attribute), 75`
`hide_error()` (`naqs-` `ICON_BUSY` (`naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`lab_devices.VISA.blacs_tab.VISATab` `attribute), 60`
`method), 12` `ICON_BUSY` (`naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`HP_8642A` (class in `naqs-` `attribute), 54`
`lab_devices.SignalGenerator.Models`), `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`
`45` `attribute), 16`
`HP_8642ATab` (class in `naqs-` `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A`), `attribute), 20`
`16` `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`HP_8642AWorker` (class in `naqs-` `attribute), 24`
`lab_devices.SignalGenerator.BLACS.HP_8642A`), `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`
`18` `attribute), 26`
`HP_8643A` (class in `naqs-` `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`
`lab_devices.SignalGenerator.Models`), `attribute), 29`
`44` `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`
`HP_8643ATab` (class in `naqs-` `attribute), 31`
`lab_devices.SignalGenerator.BLACS.HP_8643A`), `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`20` `attribute), 35`
`HP_8643AWorker` (class in `naqs-` `ICON_BUSY` (`naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.SignalGenerator.BLACS.HP_8643A`), `attribute), 39`
`22` `ICON_BUSY` (`naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`HP_8648A` (class in `naqs-` `attribute), 49`
`lab_devices.SignalGenerator.Models`),

ICON_BUSY (naqslab_devices.SR865.blacs_tab.SR865Tab attribute), 84
 lab_devices.VISA.blacs_tab.VISATab attribute), 11
 ICON_BUSY (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab attribute), 89
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab attribute), 65
 ICON_BUSY (naqslab_devices.VISA.blacs_tab.VISATab attribute), 11
 ICON_FATAL_ERROR (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab attribute), 70
 ICON_ERROR (naqs-
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab attribute), 65
 ICON_FATAL_ERROR (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab attribute), 72
 ICON_ERROR (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab attribute), 70
 ICON_FATAL_ERROR (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab attribute), 75
 ICON_ERROR (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab attribute), 72
 ICON_FATAL_ERROR (naqs-
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster attribute), 60
 ICON_ERROR (naqs-
 lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab attribute), 75
 ICON_FATAL_ERROR (naqs-
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES attribute), 54
 ICON_ERROR (naqs-
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster attribute), 60
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab attribute), 20
 ICON_ERROR (naqs-
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab attribute), 54
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab attribute), 24
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab attribute), 16
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab attribute), 24
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab attribute), 20
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab attribute), 26
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab attribute), 24
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab attribute), 29
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab attribute), 26
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab attribute), 31
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab attribute), 29
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab attribute), 35
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab attribute), 35
 ICON_FATAL_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab attribute), 49
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab attribute), 39
 ICON_FATAL_ERROR (naqs-
 lab_devices.SR865.blacs_tab.SR865Tab attribute), 84
 ICON_ERROR (naqs-
 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab attribute), 49
 ICON_FATAL_ERROR (naqs-
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab attribute), 89
 ICON_ERROR (naqs-
 lab_devices.SR865.blacs_tab.SR865Tab attribute), 84
 ICON_FATAL_ERROR (naqs-
 lab_devices.VISA.blacs_tab.VISATab attribute), 11
 ICON_ERROR (naqs-
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab attribute), 89
 ICON_OK (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab attribute), 65

ICON_OK (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BACTab (naqslab_devices.SR865.blacs_worker.SR865Worker
 attribute), 70 method), 86
 ICON_OK (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
 attribute), 73 method), 91
 ICON_OK (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab (naqslab_devices.VISA.blacs_worker.VISAWorker
 attribute), 75 method), 13
 ICON_OK (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab (naqs-
 attribute), 60 lab_devices.CounterScopeChannel method),
 ICON_OK (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
 attribute), 54 init_device_group() (naqs-
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642A_Tab (naqslab_devices.KeysightXSeries.labscrip_device.KeysightXScope
 attribute), 16 method), 69
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A_Tab (naqslab_devices.KeysightXSeries.labscrip_device.KeysightXScope
 attribute), 20 lab_devices.NovaTechDDS.labscrip_device.NovaTech409B
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648A_Tab (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech409B
 attribute), 24 init_device_group() (naqs-
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648B_Tab (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech409B
 attribute), 26 method), 80
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648C_Tab (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech440A
 attribute), 29 lab_devices.NovaTechDDS.labscrip_device.NovaTech440A
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648D_Tab (naqslab_devices.NovaTechDDS.labscrip_device.NovaTech440A
 attribute), 31 init_device_group() (naqs-
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A_Tab (naqslab_devices.PulseBlaster_No_DDS_200.labscrip_device.Pulse
 attribute), 35 method), 64
 ICON_OK (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU_Tab (naqslab_devices.PulseBlasterESRPro300.labscrip_device.PulseBl
 attribute), 39 lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBl
 ICON_OK (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab (naqslab_devices.SignalGeneratorTab method), 58
 attribute), 50 init_device_group() (naqs-
 ICON_OK (naqslab_devices.SR865.blacs_tab.SR865Tab (naqslab_devices.ScopeChannel method), 8
 attribute), 84 init_device_group() (naqs-
 ICON_OK (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab (naqslab_devices.SignalGenerator.labscrip_device.SignalGenerator
 attribute), 89 method), 54
 ICON_OK (naqslab_devices.VISA.blacs_tab.VISATab init_device_group() (naqs-
 attribute), 11 lab_devices.SignalGenerator.Models.HP_8642A
 init() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXSeriesWorker
 method), 68 init_device_group() (naqs-
 init() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BACWorker (naqslab_devices.SignalGenerator.Models.HP_8643A
 method), 77 method), 45
 init() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker (naqslab_devices.SignalGenerator.Models.HP_8648A
 method), 78 lab_devices.SignalGenerator.Models.HP_8648A
 init() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker (naqslab_devices.SignalGenerator.Models.HP_8648B
 method), 78 init_device_group() (naqs-
 init() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker (naqslab_devices.SignalGenerator.Models.HP_8648B
 method), 62 method), 47
 init() (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker (naqs-
 method), 57 lab_devices.SignalGenerator.Models.HP_8648C
 init() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker (naqslab_devices.SignalGenerator.Models.HP_8648C
 method), 19 init_device_group() (naqs-
 init() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker (naqslab_devices.SignalGenerator.Models.HP_8648D
 method), 22 method), 49
 init() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker (naqslab_devices.SignalGenerator.Models.HP_8648D
 method), 33 lab_devices.SignalGenerator.Models.HP_8648D
 init() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker (naqslab_devices.SignalGenerator.Models.RS_SMF100A
 method), 38 init_device_group() (naqs-
 init() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker (naqslab_devices.SignalGenerator.Models.RS_SMHU
 method), 41 method), 44
 init() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker (naqslab_devices.SignalGenerator.Models.RS_SMHU
 method), 52 lab_devices.SR865.labscrip_device.SR865

```

        method), 88
init_device_group() (naqs- lab_devices.StaticFreqAmp method), 10
init_device_group() (naqs- lab_devices.TektronixTDS.labscrip_device.TDS_Scope method), 93
init_device_group() (naqs- lab_devices.VISA.labscrip_device.VISA method), 15
initialise_GUI() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 65
initialise_GUI() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 70
initialise_GUI() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 74
initialise_GUI() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 74
initialise_GUI() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 75
initialise_GUI() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterES method), 61
initialise_GUI() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES method), 56
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 18
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 22
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 25
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 25
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 28
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 30
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab method), 33
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648FTab method), 36
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab method), 36
initialise_GUI() (naqs- lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 36
initialise_GUI() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 40
initialise_GUI() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 49
initialise_GUI() (naqs- lab_devices.SR865.blacs_tab.SR865Tab method), 83
initialise_GUI() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 90
initialise_GUI() (naqs- lab_devices.VISA.blacs_tab.VISATab method), 88
initialise_GUI() (naqs- lab_devices.VISA.blacs_tab.VISATab method), 11
initialise_workers() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 66
initialise_workers() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 72
initialise_workers() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 74
initialise_workers() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 76
initialise_workers() (naqs- lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster method), 61
initialise_workers() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES method), 56
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 22
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 25
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 28
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 30
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 33
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 36
initialise_workers() (naqs- lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ETab method), 36
initialise_workers() (naqs- lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 51
initialise_workers() (naqs- lab_devices.SR865.blacs_tab.SR865Tab method), 85
initialise_workers() (naqs- lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 90
initialise_workers() (naqs- lab_devices.VISA.blacs_tab.VISATab method), 88

```

`method)`, 12
`interrupt_startup()` (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker (class in naqslab_devices.KeysightXSeries.blacs_worker), `method)`, 68
`interrupt_startup()` (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker (class in naqslab_devices.NovaTechDDS.blacs_worker), `method)`, 77
`interrupt_startup()` (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_WBWorker (class in naqslab_devices.NovaTechDDS.blacs_worker), `method)`, 78
`interrupt_startup()` (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker (class in naqslab_devices.NovaTechDDS.blacs_worker), `method)`, 79
`interrupt_startup()` (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker (class in naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker), `method)`, 62
`interrupt_startup()` (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker (class in naqslab_devices.PulseBlasterESRPro300.blacs_worker), `method)`, 57
`interrupt_startup()` (naqslab_devices.SignalGenerator.BLACS.HP_8642AWorker (class in naqslab_devices.SignalGenerator.BLACS.HP_8642AWorker), `method)`, 19
`interrupt_startup()` (naqslab_devices.SignalGenerator.BLACS.HP_8643AWorker (class in naqslab_devices.SignalGenerator.BLACS.HP_8643AWorker), `method)`, 23
`interrupt_startup()` (naqslab_devices.SignalGenerator.BLACS.HP_8648Worker (class in naqslab_devices.SignalGenerator.BLACS.HP_8648Worker), `method)`, 34
`interrupt_startup()` (naqslab_devices.SignalGenerator.BLACS.RS_SMF100AWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMF100AWorker), `method)`, 38
`interrupt_startup()` (naqslab_devices.SignalGenerator.BLACS.RS_SMHUWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMHUWorker), `method)`, 42
`interrupt_startup()` (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker (class in naqslab_devices.SignalGenerator.blacs_worker), `method)`, 53
`interrupt_startup()` (naqslab_devices.SR865.blacs_worker.SR865Worker (class in naqslab_devices.SR865.blacs_worker), `method)`, 87
`interrupt_startup()` (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker (class in naqslab_devices.TektronixTDS.blacs_worker), `method)`, 91
`interrupt_startup()` (naqslab_devices.VISA.blacs_worker.VISAWorker (class in naqslab_devices.VISA.blacs_worker), `method)`, 14
`is_master_pseudoclock()` (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200 (class in naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab), `property)`, 64
`is_master_pseudoclock()` (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300 (class in naqslab_devices.PulseBlasterESRPro300.blacs_worker), `property)`, 58
K
`KeysightXScope` (class in naqslab_devices.KeysightXSeries.labscript_device), 69
`KeysightXScopeTab` (class in naqslab_devices.KeysightXSeries.labscript_device), 69

```

mainloop() (naqslab_devices.VISA.blacs_tab.VISATab
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
method), 18
mainloop() (naqslab_devices.VISA.blacs_worker.VISAWorker
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
method), 19
mainloop() (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
method), 22
mainloop() (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200Worker
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
method), 23
mainloop() (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300Worker
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648AWorker
method), 25
mainloop() (naqslab_devices.TektronixTDS.labscript_device.TDS_ScopeTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
method), 28
mainloop() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
method), 30
mainloop() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACT
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
method), 33
mainloop() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
method), 34
mainloop() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
method), 36
mainloop() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Worker
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
method), 38
mainloop() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Worker
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
method), 40
mainloop() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
method), 42
mainloop() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
method), 44
mainloop() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648AWorker
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
method), 46
mainloop() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
method), 51
mainloop() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker
lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
method), 53
mainloop() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
method), 55
mainloop() (naqslab_devices.SR865.blacs_tab.SR865Tab
lab_devices.SR865.blacs_tab.SR865Tab
method), 85
mainloop() (naqslab_devices.SR865.blacs_worker.SR865Worker
lab_devices.SR865.blacs_worker.SR865Worker
method), 87
mainloop() (naqslab_devices.VISA.blacs_tab.VISATab
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 90
mainloop() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker
lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker
method), 91
mainloop() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
method), 91

```


N

(module), 35
 n_flags (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200 (module), 39
 attribute), 63
 n_flags (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300 (module), 49
 attribute), 58
 naqslab_devices (module), 7
 naqslab_devices.KeysightXSeries (module), 52
 (module), 65
 naqslab_devices.KeysightXSeries.blacs_tab (module), 53
 (module), 65
 naqslab_devices.KeysightXSeries.blacs_worker (module), 42
 (module), 67
 naqslab_devices.KeysightXSeries.labscript_device (module), 54
 (module), 69
 naqslab_devices.KeysightXSeries.register_classes (module), 83
 (module), 70
 naqslab_devices.NovaTechDDS (module), 70
 naqslab_devices.NovaTechDDS.blacs_tab (module), 86
 (module), 70
 naqslab_devices.NovaTechDDS.blacs_worker (module), 87
 (module), 77
 naqslab_devices.NovaTechDDS.labscript_device (module), 88
 (module), 79
 naqslab_devices.NovaTechDDS.register_classes (module), 88
 (module), 83
 naqslab_devices.NovaTechDDS.runviewer_parser (module), 91
 (module), 83
 naqslab_devices.PulseBlaster_No_DDS_200 (module), 92
 (module), 59
 naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab (module), 93
 (module), 60
 naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker (module), 10
 (module), 62
 naqslab_devices.PulseBlaster_No_DDS_200.labscript_device (module), 11
 (module), 63
 naqslab_devices.PulseBlaster_No_DDS_200.register_classes (module), 13
 (module), 64
 naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser (module), 14
 (module), 64
 naqslab_devices.PulseBlasterESRPro300 (module), 15
 (module), 54
 naqslab_devices.PulseBlasterESRPro300.blacs_tab (class in naqslab_devices.NovaTechDDS.labscript_device), 80
 (module), 54
 naqslab_devices.PulseBlasterESRPro300.blacs_worker (class in naqslab_devices.NovaTechDDS.labscript_device), 79
 (module), 57
 naqslab_devices.PulseBlasterESRPro300.labscript_device (class in naqslab_devices.NovaTech409B_ACParser (class in naqslab_devices.NovaTechDDS.runviewer_parser), 83
 (module), 58
 naqslab_devices.PulseBlasterESRPro300.register_classes (class in naqslab_devices.NovaTech409B_ACTab (class in naqslab_devices.NovaTechDDS.blacs_tab), 70
 (module), 59
 naqslab_devices.PulseBlasterESRPro300.runviewer_parser (class in naqslab_devices.NovaTech409B_ACWorker (class in naqslab_devices.NovaTechDDS.blacs_worker), 77
 (module), 59
 naqslab_devices.SignalGenerator.BLACS.HP_8642A (module), 20
 (module), 16
 naqslab_devices.SignalGenerator.BLACS.HP_8643A (module), 24
 (module), 20
 naqslab_devices.SignalGenerator.BLACS.HP_8648 (module), 24
 (module), 24
 naqslab_devices.SignalGenerator.BLACS.RS_SMF100A (module), 83
 (module), 83

NovaTech409BTab (class in naqslab_devices.NovaTechDDS.blacs_tab),
72
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8642ATab,
method), 22

NovaTech409BWorker (class in naqslab_devices.NovaTechDDS.blacs_worker),
77
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab,
method), 26

NovaTech440A (class in naqslab_devices.NovaTechDDS.labscript_device),
81
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab,
method), 28

NovaTech440AParser (class in naqslab_devices.NovaTechDDS.runviewer_parser),
83
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab,
method), 30

NovaTech440ATab (class in naqslab_devices.NovaTechDDS.blacs_tab),
74
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab,
method), 33

NovaTech440AWorker (class in naqslab_devices.NovaTechDDS.blacs_worker),
78
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker,
method), 40

num_ddds (naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser.attribute), 65
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab,
method), 40

num_ddds (naqslab_devices.PulseBlasterESRPro300.runviewer_parser.attribute), 59
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab,
method), 40

num_DO (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.attribute), 60
on_force_full_buffered_reprogram()
(naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTDS_ScopeTab,
method), 85

num_DO (naqslab_devices.PulseBlasterESRPro300.blacs_tab.attribute), 54
on_force_full_buffered_reprogram()
(naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab,
method), 90

num_flags (naqslab_devices.PulseBlaster_No_DDS_200.runviewer_parser.attribute), 64
on_force_full_buffered_reprogram()
(naqslab_devices.VISA.blacs_tab.VISATab,
method), 12

num_flags (naqslab_devices.PulseBlasterESRPro300.runviewer_parser.attribute), 59
on_force_full_buffered_reprogram()
(naqslab_devices.KeysightXSeries.blacs_tab.KeysightXSScopeTab,
method), 67

offset_instructions_from_trigger() (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.method), 64
on_resolve_value_inconsistency() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXSScopeTab.method), 67

offset_instructions_from_trigger() (naqslab_devices.PulseBlasterESRPro300.labscript_device.method), 58
on_resolve_value_inconsistency() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab.method), 72

on_force_full_buffered_reprogram() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXSScopeTab.method), 67
on_resolve_value_inconsistency() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_TAB.method), 74

on_force_full_buffered_reprogram() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_TAB.method), 72
on_resolve_value_inconsistency() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab.method), 76

on_force_full_buffered_reprogram() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_TAB.method), 74
on_resolve_value_inconsistency() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterESRPro300.method), 61

on_force_full_buffered_reprogram() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab.method), 76
on_resolve_value_inconsistency() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300.method), 56

on_force_full_buffered_reprogram() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab.method), 61
on_resolve_value_inconsistency() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab.method), 18

on_force_full_buffered_reprogram() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300_Tab.method), 56
on_resolve_value_inconsistency() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab.method), 22

[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.HP_8642A](#) [method](#)), 26 [property](#)), 45
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.HP_8643A](#) [method](#)), 28 [property](#)), 45
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.HP_8648A](#) [method](#)), 30 [property](#)), 46
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.HP_8648B](#) [method](#)), 33 [property](#)), 47
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.HP_8648C](#) [method](#)), 37 [property](#)), 48
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.HP_8648D](#) [method](#)), 40 [property](#)), 49
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.RS_SMF100A](#) [method](#)), 51 [property](#)), 43
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.SR865.blacs_tab.SR865Tab](#) [parent_clock_line\(\)](#) [naqslab_devices.SignalGenerator.Models.RS_SMHU](#) [method](#)), 85 [property](#)), 44
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab](#) [parent_clock_line\(\)](#) [naqslab_devices.SR865.labscript_device.SR865](#) [method](#)), 90 [property](#)), 88
[on_resolve_value_inconsistency\(\)](#) ([naqslab_devices.VISA.blacs_tab.VISATab](#) [parent_clock_line\(\)](#) [naqslab_devices.StaticFreqAmp](#) [method](#)), 12 [property](#)), 10
P
[parent_clock_line\(\)](#) ([naqslab_devices.CounterScopeChannel](#) [parent_clock_line\(\)](#) [naqslab_devices.TektronixTDS.labscript_device.TDS_Scope](#) [property](#)), 9 [property](#)), 93
[parent_clock_line\(\)](#) ([naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope](#) [parent_clock_line\(\)](#) [naqslab_devices.VISA.labscript_device.VISA](#) [property](#)), 69 [property](#)), 15
[parent_clock_line\(\)](#) ([naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B](#) [parent_clock_line\(\)](#) [naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200](#) [property](#)), 81 [property](#)), 64
[parent_clock_line\(\)](#) ([naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B](#) [parent_clock_line\(\)](#) [naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300](#) [property](#)), 81 [property](#)), 59
[parent_clock_line\(\)](#) ([naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B](#) [parent_clock_line\(\)](#) [naqslab_devices.SR865.labscript_device.SR865](#) [property](#)), 80 [property](#)), 87
[parent_clock_line\(\)](#) ([naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A](#) [parent_clock_line\(\)](#) [naqslab_devices.SR865.blacs_worker.SR865Worker](#) [property](#)), 82 [method](#)), 86
[parent_clock_line\(\)](#) ([naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200](#) [parent_clock_line\(\)](#) [naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab](#) [property](#)), 64 [property](#)), 67
[parent_clock_line\(\)](#) ([naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300](#) [parent_clock_line\(\)](#) [naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab](#) [property](#)), 58 [property](#)), 72
[parent_clock_line\(\)](#) ([naqslab_devices.ScopeChannel](#) [parent_clock_line\(\)](#) [naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab](#) [property](#)), 8 [property](#)), 74
[parent_clock_line\(\)](#) ([naqslab_devices.SignalGenerator.labscript_device.SignalGenerator](#) [parent_clock_line\(\)](#) [naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab](#) [property](#)), 54 [property](#)), 54

140
Index

`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`, 23
`method`), 18 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`, 34
`method`), 22 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`, 38
`method`), 26 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`, 42
`method`), 28 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`, 52
`method`), 30 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.SR865.blacs_worker.SR865Worker`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`, 86
`method`), 33 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab`
`method`), 37 `program_manual()` (naqs-
`program_device_properties()` (naqs- `lab_devices.VISA.blacs_worker.VISAWorker`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker`, 13
`method`), 40 `program_static()` (naqs-
`program_device_properties()` (naqs- `lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`, 77
`method`), 51 `program_static()` (naqs-
`program_device_properties()` (naqs- `lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`
`lab_devices.SR865.blacs_tab.SR865Tab` `method`), 78
`method`), 85 `program_static()` (naqs-
`program_device_properties()` (naqs- `lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` `method`), 78
`method`), 90 `program_string` (naqs-
`program_device_properties()` (naqs- `lab_devices.SR865.blacs_worker.SR865Worker`
`lab_devices.VISA.blacs_tab.VISATab` `attribute`), 86
`method`), 12 `pseudoclock()` (naqs-
`program_manual()` (naqs- `lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS200Worker`
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`, 64
`method`), 68 `pseudoclock()` (naqs-
`program_manual()` (naqs- `lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300Worker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker`, 59
`method`), 77 `pseudoclock_device()` (naqs-
`program_manual()` (naqs- `lab_devices.CounterScopeChannel` `prop-`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`, 9
`method`), 78 `pseudoclock_device()` (naqs-
`program_manual()` (naqs- `lab_devices.KeysightXSeries.labscrip_device.KeysightXScopeWorker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker`, 70
`method`), 79 `pseudoclock_device()` (naqs-
`program_manual()` (naqs- `lab_devices.NovaTechDDS.labscrip_device.NovaTech409BWorker`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker`
`method`), 62 `pseudoclock_device()` (naqs-
`program_manual()` (naqs- `lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_ACWorker`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 57 `pseudoclock_device()` (naqs-
`program_manual()` (naqs- `lab_devices.NovaTechDDS.labscrip_device.NovaTech440AWorker`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`, 82
`method`), 19 `pseudoclock_device()` (naqs-
`program_manual()` (naqs- `lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS200Worker`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`, 64

```

pseudoclock_device() (naqs- lab_devices.PulseBlasterESRPro300.runviewer_parser),
lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
property), 59
pseudoclock_device() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab),
lab_devices.ScopeChannel property), 8 54
pseudoclock_device() (naqs- PulseBlasterESRPro300Worker (class in naqs-
lab_devices.SignalGenerator.labscript_device.SignalGenerator),
lab_devices.PulseBlasterESRPro300.blacs_worker),
property), 54 57
pseudoclock_device() (naqs- PulseblasterNoDDS200Worker (class in naqs-
lab_devices.SignalGenerator.Models.HP_8642A lab_devices.PulseBlaster_No_DDS_200.blacs_worker),
property), 45 62
pseudoclock_device() (naqs-
lab_devices.SignalGenerator.Models.HP_8643A
property), 45
pseudoclock_device() (naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B
lab_devices.SignalGenerator.Models.HP_8648A method), 81
property), 46
pseudoclock_device() (naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC
lab_devices.SignalGenerator.Models.HP_8648B method), 80
property), 47
pseudoclock_device() (naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech440A
lab_devices.SignalGenerator.Models.HP_8648C method), 82
property), 48
pseudoclock_device() (naqs- lab_devices.SignalGenerator.labscript_device.SignalGenerator
lab_devices.SignalGenerator.Models.HP_8648D method), 53
property), 49
pseudoclock_device() (naqs- lab_devices.SignalGenerator.Models.HP_8642A
lab_devices.SignalGenerator.Models.RS_SMF100A method), 45
property), 43
pseudoclock_device() (naqs- lab_devices.SignalGenerator.Models.HP_8643A
lab_devices.SignalGenerator.Models.RS_SMHU method), 45
property), 44
pseudoclock_device() (naqs- lab_devices.SignalGenerator.Models.HP_8648A
lab_devices.SR865.labscript_device.SR865 method), 46
property), 88
pseudoclock_device() (naqs- lab_devices.SignalGenerator.Models.HP_8648B
lab_devices.StaticFreqAmp property), method), 47
10
pseudoclock_device() (naqs- lab_devices.SignalGenerator.Models.HP_8648C
lab_devices.TektronixTDS.labscript_device.TDS_Scope method), 48
property), 93
pseudoclock_device() (naqs- lab_devices.SignalGenerator.Models.HP_8648D
lab_devices.VISA.labscript_device.VISA method), 49
property), 15
PulseBlaster_No_DDS_200 (class in naqs- lab_devices.SignalGenerator.Models.RS_SMF100A
lab_devices.PulseBlaster_No_DDS_200.labscript_device) method), 43
63
PulseBlaster_No_DDS_200_Parser (class in naqs- lab_devices.SignalGenerator.Models.RS_SMHU
lab_devices.PulseBlaster_No_DDS_200.runviewer_parser) method), 44
64
PulseBlaster_No_DDS_200_Tab (class in naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B
lab_devices.PulseBlaster_No_DDS_200.blacs_tab) method), 81
60
PulseBlasterESRPro300 (class in naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech409B_AC
lab_devices.PulseBlasterESRPro300.labscript_device) method), 80
58
PulseBlasterESRPro300Parser (class in naqs- lab_devices.NovaTechDDS.labscript_device.NovaTech440A
method), 82

```

quantise_freq()	(naqs- lab_devices.SignalGenerator.labscrip_device.SignalGenerator. method), 53	lab_devices.ScopeChannel method), 8
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.HP_8642A. method), 46	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.labscrip_device.SignalGenerator. method), 54
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.HP_8643A. method), 45	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.HP_8642A method), 46
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.HP_8648A. method), 46	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.HP_8643A method), 45
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.HP_8648B. method), 47	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.HP_8648A method), 46
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.HP_8648C. method), 48	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.HP_8648B method), 47
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.HP_8648D. method), 49	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.HP_8648C method), 48
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.RS_SMF100A. method), 43	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.HP_8648D method), 49
quantise_freq()	(naqs- lab_devices.SignalGenerator.Models.RS_SMHU. method), 44	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.RS_SMF100A method), 43
quantise_phase()	(naqs- lab_devices.NovaTechDDS.labscrip_device.NovaTech409B. method), 81	quantise_to_pseudoclock() (naqs- lab_devices.SignalGenerator.Models.RS_SMHU method), 44
quantise_phase()	(naqs- lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_AC. method), 80	quantise_to_pseudoclock() (naqs- lab_devices.SR865.labscrip_device.SR865 method), 88
quantise_phase()	(naqs- lab_devices.NovaTechDDS.labscrip_device.NovaTech440A. method), 82	quantise_to_pseudoclock() (naqs- lab_devices.StaticFreqAmp method), 10
quantise_to_pseudoclock()	(naqs- lab_devices.CounterScopeChannel method), 9	quantise_to_pseudoclock() (naqs- lab_devices.TektronixTDS.labscrip_device.TDS_Scope method), 93
quantise_to_pseudoclock()	(naqs- lab_devices.KeysightXSeries.labscrip_device.KeysightXScope. method), 70	quantise_to_pseudoclock() (naqs- lab_devices.VISA.labscrip_device.VISA method), 15
quantise_to_pseudoclock()	(naqs- lab_devices.NovaTechDDS.labscrip_device.NovaTech409B. method), 81	queue_work() (naqs- lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 67
quantise_to_pseudoclock()	(naqs- lab_devices.NovaTechDDS.labscrip_device.NovaTech409B_AC. method), 80	queue_work() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 72
quantise_to_pseudoclock()	(naqs- lab_devices.NovaTechDDS.labscrip_device.NovaTech440A. method), 82	queue_work() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 74
quantise_to_pseudoclock()	(naqs- lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBlaster_No_DDS_200. method), 64	queue_work() (naqs- lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 76
quantise_to_pseudoclock()	(naqs- lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300. method), 59	queue_work() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300. method), 61
quantise_to_pseudoclock()	(naqs- lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300. method), 59	queue_work() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300. method), 56
quantise_to_pseudoclock()	(naqs- lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBlasterESRPro300. method), 59	queue_work() (naqs- lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300. method), 56

`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` (naqs-
`method`), 18
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`
`method`), 22
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`
`method`), 26
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB`
`method`), 28
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648GTAB`
`method`), 30
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB`
`method`), 33
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATAB`
`method`), 37
`queue_work()` (naqs-
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTAB`
`method`), 40
`queue_work()` (naqs-
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method`), 51
`queue_work()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`method`), 85
`queue_work()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method`), 90
`queue_work()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab`
`method`), 13

R

`read_analog_parameters_string` (naqs-
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`attribute`), 67
`read_counter_string` (naqs-
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`attribute`), 68
`read_dig_parameters_string` (naqs-
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`attribute`), 67
`read_string` (naqs-
`lab_devices.SR865.blacs_worker.SR865Worker`
`attribute`), 86
`read_waveform_string` (naqs-
`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`
`attribute`), 67
`read_waveform_string` (naqs-
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`attribute`), 91
`read_x_parameters_string` (naqs-
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`attribute`), 91


```

restore_built_in_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
method), 56 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab
lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab), 22
method), 18 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab), 26
method), 22 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab), 28
method), 26 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab), 30
method), 28 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab), 33
method), 30 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab
lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab), 37
method), 33 restore_save_data() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab), 40
method), 37 restore_save_data() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab), 51
method), 40 restore_save_data() (naqslab_devices.SR865.blacs_tab.SR865Tab
lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab), 85
method), 51 restore_save_data() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
lab_devices.SR865.blacs_tab.SR865Tab), 90
method), 85 restore_save_data() (naqslab_devices.VISA.blacs_tab.VISATab
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab), 13
method), 90 RS_SMF100A (class in naqslab_devices.SignalGenerator.Models),
restore_built_in_save_data() (naqslab_devices.VISA.blacs_tab.VISATab), 42
method), 13 RS_SMF100ATab (class in naqslab_devices.SignalGenerator.BLACS.RS_SMF100A),
restore_save_data() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab5),
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab5), 67
method), 67 RS_SMF100AWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMF100A),
restore_save_data() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab),
lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab), 72
method), 72 RS_SMHU (class in naqslab_devices.SignalGenerator.Models),
restore_save_data() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab), 43
method), 74 RS_SMHUTab (class in naqslab_devices.SignalGenerator.BLACS.RS_SMHU),
restore_save_data() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab), 39
method), 76 RS_SMHUWorker (class in naqslab_devices.SignalGenerator.BLACS.RS_SMHU),
restore_save_data() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab),
lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab), 62
method), 62 run() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeV),
restore_save_data() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab), 68
method), 68 run() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_A),
lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab), 77
method), 56 run() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWo),
restore_save_data() (naqslab_devices.run() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWo),
lab_devices.run() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWo),
method), 56

```

method), 78
 run () (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker), 47
 method), 79
 run () (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker), 48
 method), 62
 run () (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker), 48
 method), 57
 run () (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker), 48
 method), 19
 run () (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker), 48
 method), 23
 run () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker), 48
 method), 34
 run () (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker), 48
 method), 38
 run () (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker), 48
 method), 42
 run () (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker), 48
 method), 53
 run () (naqslab_devices.SR865.blacs_worker.SR865Worker), 48
 method), 87
 run () (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker), 48
 method), 92
 run () (naqslab_devices.VISA.blacs_worker.VISAWorker), 48
 method), 14
 send_clear () (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker), 26
 send_clear () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648AWorker), 26
 send_clear () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker), 26
 send_clear () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CWorker), 30
 send_clear () (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DWorker), 33
 send_clear () (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker), 37
 send_clear () (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker), 41
 send_clear () (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker), 51
 send_clear () (naqslab_devices.SR865.blacs_tab.SR865Tab), 85
 send_clear () (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab), 90
 send_clear () (naqslab_devices.VISA.blacs_tab.VISATab), 11
 sens (naqslab_devices.SR865.blacs_tab.SR865Tab), 87
 sens_changed () (naqslab_devices.SR865.blacs_tab.SR865Tab), 87

S

scale_factor (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker), 19
 scale_factor (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker), 22
 scale_factor (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648AWorker), 33
 scale_factor (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BWorker), 33
 scale_factor (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker), 37
 scale_factor (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker), 41
 scale_factor (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker), 52
 scale_factor (naqslab_devices.SR865.blacs_tab.SR865Tab), 85
 scale_factor (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab), 90
 scale_factor (naqslab_devices.VISA.blacs_tab.VISATab), 11
 scale_factor (naqslab_devices.SR865.blacs_tab.SR865Tab), 87

method), 84

set_initial_trigger_time() (naqslab_devices.PulseBlaster_No_DDS_200.labscrip... method), 44

set_initial_trigger_time() (naqslab_devices.PulseBlaster_No_DDS_200.labscrip... method), 64

set_initial_trigger_time() (naqslab_devices.SR865.labscrip... method), 88

set_initial_trigger_time() (naqslab_devices.PulseBlasterESRPro300.labscrip... method), 59

set_initial_trigger_time() (naqslab_devices.StaticFreqAmp method), 10

set_phase() (naqslab_devices.SR865.labscrip... method), 87

set_phase() (naqslab_devices.TektronixTDS.labscrip... method), 93

set_properties() (naqslab_devices.CounterScopeChannel method), 9

set_properties() (naqslab_devices.VISA.labscrip... method), 15

set_properties() (naqslab_devices.KeysightXSeries.labscrip... method), 70

set_properties() (naqslab_devices.CounterScopeChannel method), 9

set_properties() (naqslab_devices.NovaTechDDS.labscrip... method), 81

set_properties() (naqslab_devices.KeysightXSeries.labscrip... method), 70

set_properties() (naqslab_devices.NovaTechDDS.labscrip... method), 80

set_properties() (naqslab_devices.NovaTech409B method), 81

set_properties() (naqslab_devices.NovaTechDDS.labscrip... method), 82

set_properties() (naqslab_devices.NovaTech440A method), 80

set_properties() (naqslab_devices.NovaTechDDS.labscrip... method), 80

set_properties() (naqslab_devices.PulseBlaster_No_DDS_200.labscrip... method), 64

set_properties() (naqslab_devices.PulseBlaster_No_DDS_200.labscrip... method), 82

set_properties() (naqslab_devices.PulseBlasterESRPro300.labscrip... method), 59

set_properties() (naqslab_devices.PulseBlasterESRPro300.labscrip... method), 64

set_properties() (naqslab_devices.ScopeChannel method), 8

set_properties() (naqslab_devices.PulseBlasterESRPro300.labscrip... method), 59

set_properties() (naqslab_devices.SignalGenerator.labscrip... method), 54

set_properties() (naqslab_devices.SignalGenerator method), 8

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8642A method), 46

set_properties() (naqslab_devices.SignalGenerator.labscrip... method), 54

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8643A method), 45

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8643A method), 46

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648A method), 46

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648A method), 45

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648B method), 47

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648B method), 47

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648C method), 48

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648B method), 47

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 49

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648C method), 48

set_properties() (naqslab_devices.SignalGenerator.Models.RS_SMF100A method), 43

set_properties() (naqslab_devices.SignalGenerator.Models.HP_8648D method), 49

set_properties() (naqslab_devices.SignalGenerator.Models.RS_SMHU method), 44

<code>lab_devices.SignalGenerator.Models.RS_SMF100A.set_tab_icon_and_colour()</code>	(naqs-
<code>method), 43</code>	<code>lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab</code>
<code>set_property()</code>	(naqs-
<code>lab_devices.SignalGenerator.Models.RS_SMHU.set_tab_icon_and_colour()</code>	(naqs-
<code>method), 44</code>	<code>lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab</code>
<code>set_property()</code>	(naqs-
<code>lab_devices.SR865.labscrip_device.SR865.set_tab_icon_and_colour()</code>	(naqs-
<code>method), 88</code>	<code>lab_devices.SR865.blacs_tab.SR865Tab</code>
<code>set_property()</code>	(naqslab_devices.StaticFreqAmp
<code>method), 10</code>	<code>method), 85</code>
<code>set_property()</code>	(naqs-
<code>lab_devices.TektronixTDS.labscrip_device.TDS_Scope</code>	<code>method), 90</code>
<code>method), 93</code>	<code>set_tab_icon_and_colour()</code>
<code>set_property()</code>	(naqs-
<code>lab_devices.VISA.labscrip_device.VISA</code>	<code>method), 13</code>
<code>method), 15</code>	<code>set_tau()</code>
<code>set_sens()</code>	(naqs-
<code>lab_devices.SR865.labscrip_device.SR865</code>	<code>method), 87</code>
<code>method), 87</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab</code>	<code>method), 67</code>
<code>method), 67</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab</code>	<code>method), 72</code>
<code>method), 72</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTAB</code>	<code>method), 74</code>
<code>method), 74</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab</code>	<code>method), 76</code>
<code>method), 76</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab</code>	<code>method), 62</code>
<code>method), 62</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab</code>	<code>method), 56</code>
<code>method), 56</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab</code>	<code>method), 18</code>
<code>method), 18</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab</code>	<code>method), 22</code>
<code>method), 22</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab</code>	<code>method), 26</code>
<code>method), 26</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTAB</code>	<code>method), 28</code>
<code>method), 28</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab</code>	<code>method), 31</code>
<code>method), 31</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTAB</code>	<code>method), 33</code>
<code>method), 33</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab</code>	<code>method), 37</code>
<code>method), 37</code>	<code>set_terminal_visible()</code>
<code>set_tab_icon_and_colour()</code>	(naqs-
<code>lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab</code>	<code>method), 41</code>

method), 41

shutdown() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorTab method), 53

set_terminal_visible() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 51

shutdown() (naqslab_devices.SR865.blacs_worker.SR865Worker method), 87

set_terminal_visible() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 85

shutdown() (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker method), 92

set_terminal_visible() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 90

shutdown() (naqslab_devices.VISA.blacs_worker.VISAWorker method), 14

set_terminal_visible() (naqslab_devices.VISA.blacs_tab.VISATab method), 13

shutdown_workers() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 67

setamp() (naqslab_devices.StaticFreqAmp method), 10

shutdown_workers() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab method), 72

setfreq() (naqslab_devices.StaticFreqAmp method), 10

shutdown_workers() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab method), 74

setphase() (naqslab_devices.StaticFreqAmp method), 10

shutdown_workers() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 76

setup_string (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker attribute), 67

shutdown_workers() (naqslab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab method), 76

setup_string (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker attribute), 91

shutdown_workers() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS200Worker method), 62

shutdown() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker method), 68

shutdown_workers() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Worker method), 56

shutdown() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTab method), 77

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab method), 18

shutdown() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker method), 78

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab method), 22

shutdown() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker method), 79

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab method), 31

shutdown() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker method), 63

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab method), 31

shutdown() (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker method), 57

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab method), 31

shutdown() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker method), 20

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab method), 33

shutdown() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker method), 23

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 37

shutdown() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 34

shutdown_workers() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab method), 38

shutdown() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 38

shutdown_workers() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 51

shutdown() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 42

shutdown_workers() (naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab method), 51

```

lab_devices.SR865.blacs_tab.SR865Tab
method), 85
lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
method), 67
shutdown_workers() (naqs- start_run() (naqs-
lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab
method), 90 method), 72
shutdown_workers() (naqs- start_run() (naqs-
lab_devices.VISA.blacs_tab.VISATab lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab
method), 13 method), 74
SignalGenerator (class in naqs- start_run() (naqs-
lab_devices.SignalGenerator.labscript_device), lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab
53 method), 76
SignalGeneratorTab (class in naqs- start_run() (naqs-
lab_devices.SignalGenerator.blacs_tab), lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster
49 method), 62
SignalGeneratorWorker (class in naqs- start_run() (naqs-
lab_devices.SignalGenerator.blacs_worker), lab_devices.PulseBlaster_No_DDS_200.blacs_worker.Pulsebla
52 method), 63
SR865 (class in naqs- start_run() (naqs-
lab_devices.SR865.labscript_device), 87 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterES
SR865Tab (class in naqs- method), 56
lab_devices.SR865.blacs_tab), 83 start_run() (naqs-
SR865Worker (class in naqs- lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlaste
lab_devices.SR865.blacs_worker), 86 method), 57
start() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker (naqs-
method), 68 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab
start() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTabWorker
method), 77 start_run() (naqs-
start() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTabWorkerSignalGenerator.BLACS.HP_8643A.HP_8643ATab
method), 78 method), 22
start() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker (naqs-
method), 79 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
start() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab
method), 62 start_run() (naqs-
start() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker
method), 63 method), 28
start() (naqslab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab (naqs-
method), 56 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
start() (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker
method), 57 start_run() (naqs-
start() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorkerSignalGenerator.BLACS.HP_8648.HP_8648DTab
method), 20 method), 33
start() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker (naqs-
method), 23 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF10
start() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker
method), 34 start_run() (naqs-
start() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorkerSignalGenerator.BLACS.RS_SMHU.RS_SMHUTab
method), 38 method), 41
start() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker (naqs-
method), 42 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
start() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker
method), 53 start_run() (naqs-
start() (naqslab_devices.SR865.blacs_worker.SR865Worker lab_devices.SR865.blacs_tab.SR865Tab
method), 87 method), 85
start() (naqslab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker (naqs-
method), 92 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
start() (naqslab_devices.VISA.blacs_worker.VISAWorker method), 90
method), 14 start_run() (naqs-
start_run() (naqs- lab_devices.VISA.blacs_tab.VISATab

```

Index	151
--------------	------------

`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 28
`method)`, 26 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` method), 31
`method)`, 28 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` method), 33
`method)`, 31 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` method), 37
`method)`, 33 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` method), 37
`method)`, 37 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` method), 51
`method)`, 41 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.SR865.blacs_tab.SR865Tab` method), 85
`method)`, 51 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab` method), 90
`method)`, 85 `statemachine_timeout_remove_all()`
`statemachine_timeout_remove()` (`naqslab_devices.VISA.blacs_tab.VISATab` method), 13
`method)`, 90 `StaticFreqAmp` (class in `naqslab_devices`), 9
`statemachine_timeout_remove()` (`naqslab_devices.VISA.blacs_tab.VISATab` method), 13
`method)`, 13 `status_byte_labels` (`naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab` attribute), 65
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` attribute), 16
`method)`, 67 `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` attribute), 20
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` attribute), 24
`method)`, 74 `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` attribute), 28
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab` attribute), 31
`method)`, 62 `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab` attribute), 33
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` attribute), 35
`method)`, 18 `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100ATab` attribute), 39
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab` attribute), 39
`method)`, 22 `status_byte_labels` (`naqslab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab` attribute), 49
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SR865.blacs_tab.SR865Tab` attribute), 49
`method)`, 26 `status_byte_labels` (`naqslab_devices.SR865.blacs_tab.SR865Tab` attribute), 49
`statemachine_timeout_remove_all()` `status_byte_labels` (`naqslab_devices.SR865.blacs_tab.SR865Tab` attribute), 49
`method)`, 26 `status_byte_labels` (`naqslab_devices.SR865.blacs_tab.SR865Tab` attribute), 49

[attribute](#)), 83
 status_byte_labels (naqs-
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 attribute), 22
 attribute), 88
 status_byte_labels (naqs-
 lab_devices.VISA.blacs_tab.VISATab
 attribute), 26
 tribute), 11
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
 attribute), 28
 method), 67
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlasterNo3DDoS_200_Tab
 attribute), 30
 method), 62
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
 lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab
 attribute), 31
 method), 56
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A
 attribute), 37
 method), 18
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A
 attribute), 41
 method), 22
 status_monitor() (naqs-
 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
 attribute), 51
 method), 26
 status_monitor() (naqs-
 lab_devices.SR865.blacs_tab.SR865Tab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
 attribute), 85
 method), 28
 status_monitor() (naqs-
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 attribute), 90
 method), 31
 status_monitor() (naqs-
 lab_devices.VISA.blacs_tab.VISATab
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
 attribute), 11
 method), 33
 status_monitor() (naqs-
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A
 attribute), 65
 method), 37
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642A
 lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU
 attribute), 16
 method), 41
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A
 lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab
 attribute), 20
 method), 51
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab
 lab_devices.SR865.blacs_tab.SR865Tab
 attribute), 24
 method), 85
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab
 lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab
 attribute), 26
 method), 90
 status_monitor() (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab
 lab_devices.VISA.blacs_tab.VISATab
 attribute), 29
 method), 11
 status_widget (naqs-
 lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab
 lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab
 attribute), 31
 attribute), 67
 status_widget (naqs-
 lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A
 attribute), 35
 attribute), 18

method), 51

supports_smart_programming() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 86

supports_smart_programming() (naqslab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab method), 91

supports_smart_programming() (naqslab_devices.VISA.blacs_tab.VISATab method), 13

T

t0() (naqslab_devices.CounterScopeChannel property), 9

t0() (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScopeTab property), 70

t0() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech409B.blacs_worker.NovaTech409BWorker method), 81

t0() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A.blacs_worker.NovaTech440AWorker method), 80

t0() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A.blacs_worker.NovaTech440AWorker method), 79

t0() (naqslab_devices.NovaTechDDS.labscript_device.NovaTech440A.blacs_worker.NovaTech440AWorker method), 82

t0() (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker method), 64

t0() (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker method), 59

t0() (naqslab_devices.ScopeChannel property), 8

t0() (naqslab_devices.SignalGenerator.labscript_device.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 54

t0() (naqslab_devices.SignalGenerator.Models.HP_8642A.blacs_worker.SignalGeneratorWorker method), 46

t0() (naqslab_devices.SignalGenerator.Models.HP_8643A.blacs_worker.SignalGeneratorWorker method), 45

t0() (naqslab_devices.SignalGenerator.Models.HP_8648A.blacs_worker.SignalGeneratorWorker method), 47

t0() (naqslab_devices.SignalGenerator.Models.HP_8648B.blacs_worker.SignalGeneratorWorker method), 47

t0() (naqslab_devices.SignalGenerator.Models.HP_8648C.blacs_worker.SignalGeneratorWorker method), 48

t0() (naqslab_devices.SignalGenerator.Models.HP_8648D.blacs_worker.SignalGeneratorWorker method), 49

t0() (naqslab_devices.SignalGenerator.Models.RS_SMF100A.blacs_worker.SignalGeneratorWorker method), 43

t0() (naqslab_devices.SignalGenerator.Models.RS_SMHU.blacs_worker.SignalGeneratorWorker method), 44

t0() (naqslab_devices.SR865.labscript_device.SR865.blacs_worker.SR865Worker method), 88

t0() (naqslab_devices.StaticFreqAmp property), 10

t0() (naqslab_devices.TektronixTDS.labscript_device.TDS_ScopeTab property), 93

t0() (naqslab_devices.VISA.labscript_device.VISA.blacs_worker.VISAWorker method), 15

tau (naqslab_devices.SR865.labscript_device.SR865 attribute), 87

tau_changed() (naqslab_devices.SR865.blacs_tab.SR865Tab method), 83

TDS_Scope (class in naqslab_devices.TektronixTDS.labscript_device), 92

TDS_ScopeTab (class in naqslab_devices.TektronixTDS.blacs_tab), 88

TDS_ScopeWorker (class in naqslab_devices.TektronixTDS.blacs_worker), 91

terminate() (naqslab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker method), 69

terminate() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWorker method), 77

terminate() (naqslab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker method), 78

terminate() (naqslab_devices.NovaTech440A.blacs_worker.NovaTech440AWorker method), 79

terminate() (naqslab_devices.NovaTech440A.blacs_worker.NovaTech440AWorker method), 82

terminate() (naqslab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS_200Worker method), 63

terminate() (naqslab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker method), 57

terminate() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 20

terminate() (naqslab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker method), 23

terminate() (naqslab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker method), 23

terminate() (naqslab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker method), 34

terminate() (naqslab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker method), 38

terminate() (naqslab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker method), 42

terminate() (naqslab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker method), 53

terminate() (naqslab_devices.SR865.blacs_worker.SR865Worker method), 87

transition_to_buffered() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 67

transition_to_buffered() (naqslab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab method), 67

`lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWorker`), 35
`method`), 68 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`), 37
`method`), 72 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` `method`), 38
`method`), 74 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` `method`), 41
`method`), 76 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWo`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACTab`), 42
`method`), 77 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`), 51
`method`), 78 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWo`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker`), 52
`method`), 79 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 62 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.SR865.blacs_worker.SR865Worker`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker`
`method`), 63 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 57 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 57 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab`), 13
`method`), 18 `transition_to_buffered()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.VISA.blacs_worker.VISAWorker`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`), 14
`method`), 20 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab`), 67
`method`), 22 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.KeysightXSeries.blacs_worker.KeysightXScopeWo`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`), 68
`method`), 24 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab`), 72
`method`), 26 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab`), 74
`method`), 29 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648CTab`), 77
`method`), 31 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACWo`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648DTab`), 77
`method`), 33 `transition_to_manual()` (`naqs-`
`transition_to_buffered()` (`naqs-` `lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker`), 77

`transition_to_manual()` (naqs- `lab_devices.SignalGenerator.blacs_worker.SignalGeneratorWorker`
`lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWWorker`), 53
`method`), 79 `transition_to_manual()` (naqs-
`lab_devices.SR865.blacs_tab.SR865Tab`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200_Tab`
`method`), 62 `transition_to_manual()` (naqs-
`lab_devices.SR865.blacs_worker.SR865Worker`
`lab_devices.PulseBlaster_No_DDS_200.blacs_worker.PulseBlaster_No_DDS200Worker`
`method`), 63 `transition_to_manual()` (naqs-
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab`
`method`), 57 `transition_to_manual()` (naqs-
`lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`lab_devices.PulseBlasterESRPro300.blacs_worker.PulseBlasterESRPro300Worker`
`method`), 57 `transition_to_manual()` (naqs-
`lab_devices.VISA.blacs_tab.VISATab`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642A_Tab`), 13
`method`), 18 `transition_to_manual()` (naqs-
`lab_devices.VISA.blacs_worker.VISAWorker`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642AWorker`
`method`), 20 `trigger()` (naqslab_devices.KeysightXSeries.labscript_device.KeysightXScope
`method`), 70
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643A_Tab` (naqslab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200
`method`), 22 `method`), 64
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643AWorker`
`method`), 24 `trigger()` (naqslab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300
`method`), 59
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648_Tab`
`method`), 26 `trigger()` (naqslab_devices.TektronixTDS.labscript_device.TDS_Scope
`method`), 93
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648_Tab` (naqs-
`method`), 29 `attribute`), 64
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648_Tab` (naqs-
`method`), 29 `attribute`), 59
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648_Tab` (naqs-
`method`), 31 `lab_devices.TektronixTDS.labscript_device.TDS_Scope`
`attribute`), 92
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648_Tab` (naqs-
`method`), 33 `lab_devices.KeysightXSeries.labscript_device.KeysightXScope`
`attribute`), 70
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648Worker` (naqs-
`method`), 35 `lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200`
`attribute`), 64
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100A_Tab` (naqs-
`method`), 37 `lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300`
`attribute`), 59
`lab_devices.SignalGenerator.BLACS.RS_SMF100A.RS_SMF100AWorker` (naqs-
`method`), 39 `lab_devices.TektronixTDS.labscript_device.TDS_Scope`
`attribute`), 93
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHU_Tab` (naqs-
`method`), 41 `lab_devices.PulseBlaster_No_DDS_200.labscript_device.PulseBlaster_No_DDS_200`
`attribute`), 64
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUWorker` (naqs-
`method`), 42 `lab_devices.PulseBlasterESRPro300.labscript_device.PulseBlasterESRPro300`
`attribute`), 59
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method`), 51
`transition_to_manual()` (naqs- `update_from_settings()` (naqs-

`lab_devices.KeysightXSeries.blacs_tab.KeysightXScopeTab1`
`method), 67` `VISAWorker` (class in `naqs-`
`update_from_settings()` (`naqs-` `lab_devices.VISA.blacs_worker`), 13
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409B_ACTab`
`method), 72`
`update_from_settings()` (`naqs-` `wait_delay` (`naqs-`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech409BTab` `lab_devices.PulseBlaster_No_DDS_200.labscrip_device.PulseBl`
`method), 74` `attribute`), 64
`update_from_settings()` (`naqs-` `wait_delay` (`naqs-`
`lab_devices.NovaTechDDS.blacs_tab.NovaTech440ATab` `lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBl`
`method), 77` `attribute`), 59
`update_from_settings()` (`naqs-` `waveform_parser()` (`naqs-`
`lab_devices.PulseBlaster_No_DDS_200.blacs_tab.PulseBlaster_No_DDS_200Tab` `lab_devices.TektronixTDS.blacs_worker.TDS_ScopeWorker`
`method), 62` `method`), 91
`update_from_settings()` (`naqs-` `write_check()` (`naqs-`
`lab_devices.PulseBlasterESRPro300.blacs_tab.PulseBlasterESRPro300Tab` `lab_devices.NovaTechDDS.blacs_worker.NovaTech409B_ACW`
`method), 57` `method`), 77
`update_from_settings()` (`naqs-` `write_check()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8642A.HP_8642ATab` `lab_devices.NovaTechDDS.blacs_worker.NovaTech409BWorker`
`method), 18` `method`), 78
`update_from_settings()` (`naqs-` `write_check()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8643A.HP_8643ATab` `lab_devices.NovaTechDDS.blacs_worker.NovaTech440AWorker`
`method), 22` `method`), 79
`update_from_settings()` (`naqs-` `write_pb_inst_to_h5()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648ATab` `lab_devices.PulseBlaster_No_DDS_200.labscrip_device.Pulse`
`method), 26` `method`), 64
`update_from_settings()` (`naqs-` `write_pb_inst_to_h5()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.HP_8648.HP_8648BTab` `lab_devices.PulseBlasterESRPro300.labscrip_device.PulseBl`
`method), 29` `method`), 59
`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.RS_SMF100A.RS_SMF100ATab`
`method), 37`
`update_from_settings()` (`naqs-`
`lab_devices.SignalGenerator.BLACS.RS_SMHU.RS_SMHUTab`
`method), 41`
`update_from_settings()` (`naqs-`
`lab_devices.SignalGenerator.blacs_tab.SignalGeneratorTab`
`method), 52`
`update_from_settings()` (`naqs-`
`lab_devices.SR865.blacs_tab.SR865Tab`
`method), 86`
`update_from_settings()` (`naqs-`
`lab_devices.TektronixTDS.blacs_tab.TDS_ScopeTab`
`method), 91`
`update_from_settings()` (`naqs-`
`lab_devices.VISA.blacs_tab.VISATab`
`method), 13`

V

`VISA` (class in `naqs-`
`lab_devices.VISA.labscrip_device`), 14
`VISATab` (class in `naqslab_devices.VISA.blacs_tab`),