ptb-drivers Documentation

Release 0.1

Robert Jördens

May 31, 2018

Contents

1 Synthesizer **3**
- 1.1 `ptb.synth_protocol` module . . . 3
- 1.2 `ptb.synth_tcp` module . . . 3
- 1.3 `ptb.adf4350` module . . . 4

2 Temperature Sensor **5**
- 2.1 `ptb.temp_protocol` module . . . 5
- 2.2 `ptb.temp_tcp` module . . . 5

3 Voltage Source **7**
- 3.1 `ptb.voltage_protocol` module . . . 7
- 3.2 `ptb.voltage_tcp` module . . . 8

4 Indices and tables **9**

Python Module Index **11**

Contents:

CHAPTER 1

Synthesizer

1.1 ptb.synth_protocol module

class ptb.synth_protocol.**SynthProtocol**
Protocol for the PTB synthesizer (ADF4350-based)

save(*regs=None*)
Save the six registers to the EEPROM. That data is loaded on boot of the synthesizer.

Args:

regs (list(int), optional): 32 bit register values (reg5 down to reg0). If no register values are passed, then the ones calculated by `set_frequency()` are used.

start(*regs=None*)
Send the six registers to the synthesizer.

Args:

regs (list(int), optional): 32 bit register values (reg5 down to reg0). If no register values are passed, then the ones calculated by `set_frequency()` are used.

version()
Return the hardware/firmware version.

Returns: str: Version string.

1.2 ptb.synth_tcp module

class ptb.synth_tcp.**SynthTCP**(*reader*, *writer*)

1.3 ptb.adf4350 module

class ptb.adf4350.**ADF4350**

From the dtasheet with analogies of naming from https://github.com/analogdevicesinc/no-OS/blob/master/drivers/adf4350/adf4350.h

set_frequency(*f_out*)

Set output frequency.

Args: f_out (float): Desired frequency

Returns: Actual output frequency set

CHAPTER 2

Temperature Sensor

2.1 ptb.temp_protocol module

class ptb.temp_protocol.**TempProtocol**
Protocol for the PTB multi-channel temperature sensor

get(*channel*)
Measure the temperature on one channel.

Args: channel (int): Channel index to measure on

Return: float: Temperature

get_all()
Measure and return the temperatures on all channels.

Returns: list(float): Temperatures on all channels

version()
Return the hardware/firmware version.

Returns: str: Version string.

2.2 ptb.temp_tcp module

class ptb.temp_tcp.**TempTCP**(*reader*, *writer*)

CHAPTER 3

Voltage Source

3.1 ptb.voltage_protocol module

class ptb.voltage_protocol.**VoltageProtocol**
Protocol for the PTB multi-channel voltage source

factory()
Reset gains and offsets to default values

get_data(*channels=None*)
Get raw channel output values. Values are given in DAC LSBs (integers).

Args: channels (list(int)): Target channels. Defaults to 1...8

Returns: list(int)): DAC values, one for each target channel.

get_temperature()
Get temperature data for controller and amplifier boards.

Returns: list(float): Raw ADC values for the NTCs

ldac()
Pulse LDAC to all DACs to load values into active registers.

set_data(*values*, *channels=None*)
Set raw channel output values. Values are given in DAC LSBs (integers). Data becomes active only after *ldac()*.

Args: values (list(int)): DAC values, one for each target channel. channels (list(int)): Target channels. Defaults to 1...len(values)

Returns: list(int): Actual values returned by the device.

set_gain(*values*, *channels=None*)
Set channel gains. Channel gains are processed within the microcontroller and become active on set_volt() and a subsequent *ldac()*.

Gains are given in *2**16/full_scale* where *full_scale = u_max - u_min*.

Args: values (list(float)): Gains, one for each target channel. channels (list(int)): Target channels. Defaults to 1... len(values)

Returns: list(float): Actual values returned by the device.

set_offset(*values*, *channels=None*)

Set channel offsets. Channel gains are processed within the microcontroller and become active on `set_volt()` and a subsequent *ldac()*.

Offsets are given in DAC LSBs (integers).

Args: values (list(int)): Offsets, one for each target channel. channels (list(int)): Target channels. Defaults to 1... len(values)

Returns: list(int): Actual values returned by the device.

set_voltage(*values*, *channels=None*)

Set output voltages. Voltages become active only after *ldac()*.

Args: values (list(float)): Voltages, one for each target channel. channels (list(int)): Target channels. Defaults to 1... len(values)

Returns: list(float): Actual values returned by the device.

version()

Return the hardware/firmware version.

Returns: str: Version string.

3.2 ptb.voltage_tcp module

class ptb.voltage_tcp.**VoltageTCP**(*reader*, *writer*)

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

ptb.adf4350, 4
ptb.synth_protocol, 3
ptb.synth_tcp, 3
ptb.temp_protocol, 5
ptb.temp_tcp, 5
ptb.voltage_protocol, 7
ptb.voltage_tcp, 8

Index

A

ADF4350 (class in ptb.adf4350), 4

F

factory() (ptb.voltage_protocol.VoltageProtocol method), 7

G

get() (ptb.temp_protocol.TempProtocol method), 5
get_all() (ptb.temp_protocol.TempProtocol method), 5
get_data() (ptb.voltage_protocol.VoltageProtocol method), 7
get_temperature() (ptb.voltage_protocol.VoltageProtocol method), 7

L

ldac() (ptb.voltage_protocol.VoltageProtocol method), 7

P

ptb.adf4350 (module), 4
ptb.synth_protocol (module), 3
ptb.synth_tcp (module), 3
ptb.temp_protocol (module), 5
ptb.temp_tcp (module), 5
ptb.voltage_protocol (module), 7
ptb.voltage_tcp (module), 8

S

save() (ptb.synth_protocol.SynthProtocol method), 3
set_data() (ptb.voltage_protocol.VoltageProtocol method), 7
set_frequency() (ptb.adf4350.ADF4350 method), 4
set_gain() (ptb.voltage_protocol.VoltageProtocol method), 7
set_offset() (ptb.voltage_protocol.VoltageProtocol method), 8
set_voltage() (ptb.voltage_protocol.VoltageProtocol method), 8
start() (ptb.synth_protocol.SynthProtocol method), 3
SynthProtocol (class in ptb.synth_protocol), 3
SynthTCP (class in ptb.synth_tcp), 3

T

TempProtocol (class in ptb.temp_protocol), 5
TempTCP (class in ptb.temp_tcp), 5

V

version() (ptb.synth_protocol.SynthProtocol method), 3
version() (ptb.temp_protocol.TempProtocol method), 5
version() (ptb.voltage_protocol.VoltageProtocol method), 8
VoltageProtocol (class in ptb.voltage_protocol), 7
VoltageTCP (class in ptb.voltage_tcp), 8