**hyperdaq.card**

card.py

Optimized hyperDAQ modules for talking to National Instruments Data Acquisition Card

Replacing nidaq.py and controls.py

Last Updated: January 2020

Trevor Arp

Gabor Lab

University of California, Riverside

All Rights Reserved

**hyperdaq.card.CAL(*val*, *channel*)**

Calibrate Voltage Channels

***class*hyperdaq.card.card\_control\_acquire(*num\_queues*)**

Bases: **object**

Controls the National Instruments (NI) DAQ card for the purpose of acquiring data from scanning measurements.

Multiple queues facilitates sending data into multiple places, for example a data writing queue, a data analysis queue and a display queue. The function get\_queues returns a list containing all the queuesData goes into queues as an array with the 1st column being time and the other columns containing voltage values of the channels.

CURRENT IMPLEMENTATION is for four voltage channels, may be more extensive in the future. Channels are defined (e.g. ‘Dev1/ao0’) in the parameters file: CHANNEL\_x, CHANNEL\_y, CHANNEL\_v1, CHANNEL\_v2

**Parameters**

**num\_queues** (*int*) – The number of Queues that the data is being written into.

**nx**

the number of x points in the image, by default NUM\_x\_points. READ ONLY

**ny**

the number of y points in the image, by default NUM\_y\_points. READ ONLY

**linerate**

the number of scanned lines per second, when scanning. READ ONLY

**card\_output\_waveform\_scan(*x\_range*, *y\_range*, *vsd\_range*, *vbg\_range*, *theta*, *func\_range=(0*, *0)*, *func=None*, *after\_vbg=False*, *after\_vsd=False*)**

Scans continuously along a fast axis and in discrete steps along an orthogonal slow axis, outputting a scanning waveform along the fast axis.

xrange and yrange are tuples containing (vstart, vend) where the scan is from vstart to vend, theta the scan angle along the fast and slow axes respectively, if vstart = vend there will be no scan along that axis, instead that axis will stay at a constant value given by vstart=vend. Outputs will be zeroed at the end of the scan, unless default value is set

All input parameter voltages will be calibrated such that they give the correct output value at the physical output, calibration parameters are defined for each channel in the parameters file.

Also scans the input function func along the slow axis in range funcrange. If funcrange is a tuple of length 2 or a numpy array then the input function should be callable with a float value that changes some parameter, i.e. func(x).

after\_vbg if True the backgate will stay constant at the ending value after the scan is sucessfully Completed, if False it will go to the default value

after\_vsd if True the backgate will stay constant at the ending value after the scan is sucessfully Completed, if False it will go to the default value

**clear\_queues()**

Empties the data Queues

**compute\_waveform(*startV*, *stopV*, *nextV*, *N*)**

Computes the waveform needed to scan over 80 percent of the waveform, ramping back to the next point over the remaining 20 percent of the waveform

where startV and stopV are the points to ramp between and nextV is the point to ramp down to after reaching stopV, all must be floats

N is the number of points to do it for, must be an integer

**est\_scan\_time()**

Estimates the number of seconds needed to scan

**fixed\_output\_line\_scan(*x\_range*, *y\_range*, *vsd\_range*, *vbg\_range*, *theta*, *fast\_func=None*, *fast\_func\_range=(0*, *0)*, *func=None*, *func\_range=(0*, *0)*, *after\_vbg=False*, *after\_vsd=False*)**

Scans in discrete steps along the fast and slow axes, with all the card outputs (space, voltage) constant on each line (i.e. only being slow variables). fast\_func will be called as the fast axis, if None it will throw an error.

xrange and yrange are tuples containing (vstart, vend) where the scan is from vstart to vend, theta the scan angle along the fast and slow axes respectively, if vstart = vend there will be no scan along that axis, instead that axis will stay at a constant value given by vstart=vend. Outputs will be reset to their default values (0 unless specified otherwise) at the end of the scan.

All input parameter voltages will be calibrated such that they give the correct output value at the physical output, calibration parameters are defined for each channel in the parameters file.

Also scans the input function func along the slow axis in range funcrange. If funcrange is a tuple of length 2 or a numpy array then the input function should be callable with a float value that changes some parameter, i.e. func(x). If funcrange is a tuple of length 4, then the input function should be callable with three float inputs, two of which are constant and the third that changes some parameter, i.e. func(funcrange[2], funcrange[3], x).

after\_vbg if True the backgate will stay constant at the ending value after the scan is sucessfully Completed, if False it will go to the default value

after\_vsd if True the backgate will stay constant at the ending value after the scan is sucessfully Completed, if False it will go to the default value

**get\_output\_queues()**

Returns a list containing all the Queues that the data is being written into

**get\_queues()**

Returns point Queues for processing

**pollcard(*N=2000*)**

Grabs the current values of the inputs and sends it off for processing

**ramp\_up(*X*, *Y*, *Vsd*, *Vbg*, *t=-1*)**

Ramps from the current value to the desired value for each channel, in the time defined by the parameter t which if set to -1 is the default CNTL\_default\_ramp\_time

**reset\_input()**

Re-sets the input channels, clearing out any registered functions, allocated resources, etc. Should be called after scanning functions are finished or aborted.

**scan(*type*, *outputs*, *fastchannel*, *\*\*kwargs*)**

Wrapper function for all the types of scans implemented, based on type will pass arguments to the appropriate scanning function.

Implemented scan types: - “rectilinear-4output”: A basic rectilinear scan based on a card with 4 voltage outputs designated (xaxis, yaxis, vsd, vbg)

fastchannel sets which output channel is the fast channel, if it doesn’t match specific cases it will be the x-axis

Included for extensibility, additional types of scans may become available in the future.

**set\_default\_voltage(*chan*, *v*)**

Sets and ramps to the default value of source/drain or backgate

**set\_line\_rate(*rate*)**

Sets the line rate, i.e. the rate of scanning for the fast axis, in lines/second.

**set\_nxy(*nx*, *ny*)**

Set the size of the data image

**stop()**

Closes the voltage ouput channels, sets them to zero

**zero\_all()**

Zeros all voltage channels