

Table of correlated SCPI and API commands on Red Pitaya (date: 3.3.2016 , 13.00)

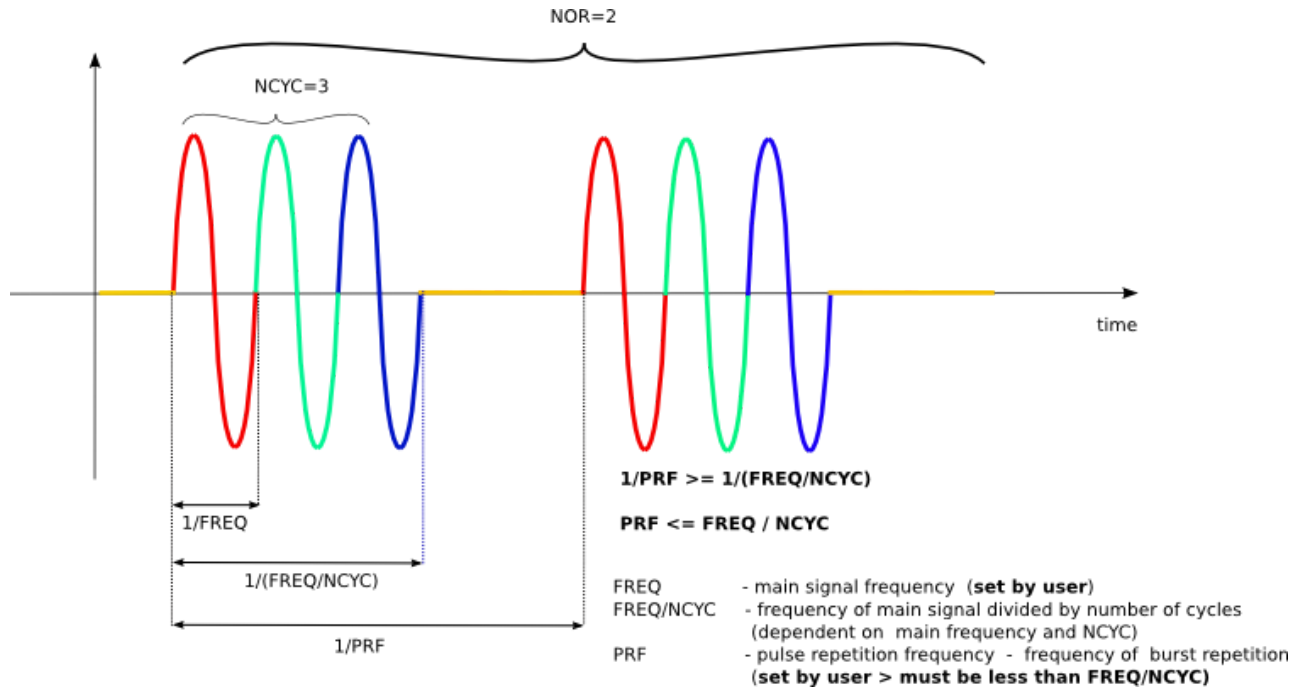
SCPI	OPTIONS	DESCRIPTION	API
LED diodes and GPIOs Red Pitaya			
DIG:PIN:DIR <dir>,<pin> Examples: DIG:PIN:DIR OUT,DIO0_N DIG:PIN:DIR IN,DIO1_P	<dir> = {OUT,IN} <pin>={DIO1_P...DIO7_P, DIO0_N...DIO7_N} OUT = OUTPUT IN = INPUT Default: IN	Set direction of digital pins to output or input.	rp_DpinSetDirection
DIG:PIN <pin>,<state> Examples: DIG:PIN DIO0_N,1 DIG:PIN LED2,1	<pin>={DIO1_P...DIO7_P, DIO0_N...DIO7_N, LED1...LED8} <state>={0,1} Default: 0	Set state of digital outputs to 1(HIGH) or 0(LOW).	rp_DpinSetState
DIG:PIN? <pin> Examples: DIG:PIN? DIO0_N DIG:PIN? LED2 Query return: {0, 1, ERR}	<pin>={DIO1_P...DIO7_P, DIO0_N...DIO7_N, LED1...LED8}	Get state of digital inputs and outputs.	rp_DpinGetState
Analog Inputs and Outputs			
ANALOG:PIN <pin>,<value> Examples: ANALOG:PIN AOUT0,1 ANALOG:PIN AOUT2,1.34	<pin>={AOUT0, AOUT1, AOUT2, AOUT3} <value>={value in Volts} Default: 0	Set analog voltage on slow analog outputs. Voltage range of slow analog outputs is: 0 -1.8 V	rp_ApinSetValue
ANALOG:PIN? <pin> Examples: ANALOG:PIN? AOUT0 ANALOG:PIN? AIN2 Query return: {value in Volts, ERR}	<pin>={AIN0, AIN1, AIN2, AIN3, AOUT0, AOUT1, AOUT2, AOUT3}	Read analog voltage from slow analog inputs. Voltage range of slow analog inputs is: 0 -3.3 V	rp_ApinGetValue
Signal Generator			

<n> = {1,2} (set channel OUT1 or OUT2)

<p>OUTPUT<n>:STATE <par></p> <p>Examples: OUTPUT1:STATE ON OUTPUT2:STATE OFF</p>	<p><par>={ON,OFF}</p> <p>Default: OFF</p>	<p>Disable or enable fast analog outputs.</p>	<p>rp_GenOutEnable rp_GenOutDisable</p>
<p>SOUR<n>:FREQ:FIX <value></p> <p>Examples: SOUR1:FREQ:FIX 1000 SOUR2:FREQ:FIX 100000</p>	<p><value>={frequency 0Hz-62.5e6Hz}</p> <p>Default: 1000</p>	<p>Set frequency of fast analog outputs.</p>	<p>rp_GenFreq</p>
<p>SOUR<n>:FUNC <par></p> <p>Examples: SOUR1:FUNC SINE SOUR2:FUNC TRIANGLE</p>	<p><par>={SINE, SQUARE, TRIANGLE, SAWU, SAWD, PWM, ARBITRARY}</p> <p>Default: SINE</p>	<p>Set waveform of fast analog outputs.</p>	<p>rp_GenWaveform</p>
<p>SOUR<n>:VOLT <value></p> <p>Examples: SOUR1:VOLT 1 SOUR2:VOLT 0.5</p>	<p><value>={amplitude -1V - 1V}</p> <p>Default: 1</p> <p>AMP+OFFS <= 1 V</p>	<p>Set amplitude voltage of fast analog outputs. Amplitude + offset value must be less than maximum output range +/- 1V</p>	<p>rp_GenAmp</p>
<p>SOUR<n>:VOLT:OFFS <value></p> <p>Examples: SOUR1:VOLT:OFFS 0.2 SOUR1:VOLT:OFFS 0.1</p>	<p><value>={offset -1V - 1V}</p> <p>Default: 0</p> <p>AMP+OFFS <= 1 V</p>	<p>Set offset voltage of fast analog outputs. Amplitude + offset value must be less than maximum output range +/- 1V</p>	<p>rp_GenOffset</p>
<p>SOUR<n>:PHAS <value></p> <p>Examples: SOUR2:PHAS 30</p>	<p><value>={phase -360deg - 360deg}</p> <p>Default: 0</p>	<p>Set phase of fast analog outputs.</p>	<p>rp_GenPhase</p>
<p>SOUR<n>:DCYC <par></p> <p>Examples: SOUR1:DCYC 34 SOUR2:DCYC 50</p>	<p><value>={duty cycle 0-100}</p> <p>Default: 50</p> <p>Only for PWM</p>	<p>Set duty cycle of PWM waveform.</p>	<p>rp_GenDutyCycle</p>
<p>SOUR<n>:TRAC:DATA:DATA <array></p> <p>Examples:</p>	<p><array>={value1, value2, ...valueN}</p> <p>max. 16k values</p>	<p>Import data for arbitrary waveform generation.</p>	<p>rp_GenArbWaveform</p>

SOUR1:TRAC:DATA:DATA 1,0.5,0.2	Values are floats in range from -1 to 1.		
SOUR<n>:BURS:STAT <par> Examples: SOUR1:BURS:STAT ON SOUR1:BURS:STAT OFF	<par>={ON,OFF} Default: OFF	Enable or disable burst (pulse) mode. Red Pitaya will generate R-times N periods of signal and then stop. Time between is P.	rp_GenMode
SOUR<n>:BURS:NCYC <value> Examples: SOUR1:BURS:NCYC 3	<value>={burst count 1-50000, INF} INF = infinity - continuous Default: 1	Set N number of generated signals in one burst	rp_GenBurstCount
SOUR1:BURS:NOR <value> Examples: SOUR1:BURS:NOR 5	<value>={burst repetitions 1-50000, INF} INF = infinity	Set R number of repeated bursts	rp_GenBurstRepetitions
SOUR1:BURS:INT:PER <value> Examples: SOUR1:BURS:INT:PER 1000000	<value>={burst period 1us-500s}	Set P total time of one burst in micro seconds. This includes the signal and delay	rp_GenBurstPeriod
SOUR<n>:TRIG:SOUR <par> Examples: SOUR1:TRIG:SOUR EXT	<par>={EXT_PE,EXT_NE,INT, GATED} EXT = External INT = Internal GATED = gated busts Default: INT	Set trigger source for selected signal.	rp_GenTriggerSource
SOUR<n>:TRIG:IMM Examples: SOUR1:TRIG:IMM		Triggers selected source immediately	rp_GenTrigger
TRIG:IMM Examples: TRIG:IMM		Triggers both sources immediately	rp_GenTrigger
GEN:RST		Reset generator to default settings.	

Examples:
GEN:RST



Acquire

<n> = {1,2} (set channel IN1 or IN2)

Control

ACQ: START

Examples:
ACQ: START

Starts acquisition.

rp_AcqStart

ACQ: STOP

Examples:
ACQ: STOP

Stops acquisition.

rp_AcqStop

ACQ: RST

Examples:
ACQ: STOP

Stops acquisition and sets all parameters to default values.

rp_AcqReset

Sampling rate & decimation

ACQ: DEC <par>

<par>={1, 8, 64, 1024, 8192, 65536}

Default: 1

Set decimation factor.

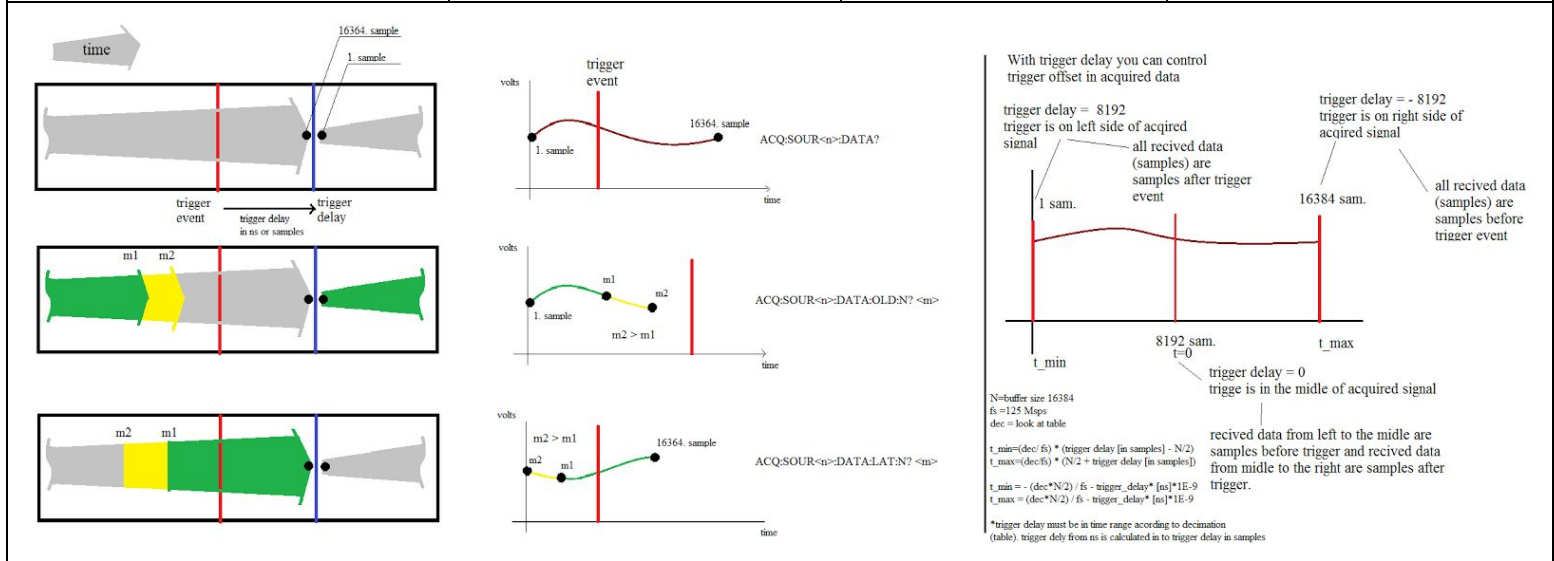
rp_AcqSetDecimation

<p>ACQ:DEC?</p> <p>Example: ACQ:DEC?</p> <p>Query return: {1,8,64,1024,8192,65536}</p>		Get decimation factor.	rp_AcqGetDecimation
	<p><par>={125MHz,15_6MHz, 1_9MHz,103_8kHz, 15_2kHz, 1_9kHz}</p> <p>Default: 125MHz</p>	Set sampling rate.	rp_AcqSetSamplingRate
<p>ACQ:SRAT?</p> <p>Example: ACQ:SRAT?</p> <p>Query return: {125MHz,15_6MHz, 1_9MHz,103_8kHz, 15_2kHz, 1_9kHz}</p>		Get sampling rate.	rp_AcqGetSamplingRate
<p>ACQ:SRA:HZ?</p> <p>Example: ACQ:SRA:HZ?</p> <p>Query return: 125000000 Hz</p>		Get sampling rate in Hz.	rp_AcqGetSamplingRateHz
<p>ACQ:AVG <par></p>	<p><par>={OFF,ON}</p> <p>Default: ON</p>	Enable/disable averaging.	rp_AcqSetAveraging
<p>ACQ:AVG?</p> <p>Example: ACQ:AVG?</p> <p>Query return: {OFF,ON}</p>		Get averaging status.	rp_AcqGetAveraging
Trigger			
<p>ACQ:TRIG <par></p> <p>Example: ACQ:TRIG CH1_PE</p>	<p><par>={DISABLED,NOW,CH1_P E,CH1_NE,CH2_PE,CH2_NE,EX T_PE,EXT_NE,AWG_PE, AWG_NE}</p> <p>Default: DISABLED</p>	Disable triggering, trigger immediately or set trigger source & edge.	rp_AcqSetTriggerSrc

<p>ACQ:TRIG:STAT?</p> <p>Example: ACQ:TRIG:STAT?</p> <p>Query return: {WAIT,TD}</p>		<p>Get trigger status.</p>	<p>rp_AcqGetTriggerState</p> <p>if DISABLED -> TD else WAIT</p>
<p>ACQ:TRIG:DLY <par></p> <p>Example: ACQ:TRIG:DLY 2314</p>	<p><par>={value in samples}</p> <p>Default: 0</p>	<p>Set trigger delay in samples.</p>	<p>rp_AcqSetTriggerDelay</p>
<p>ACQ:TRIG:DLY?</p> <p>Example: ACQ:TRIG:DLY?</p> <p>Query return: 2314</p>		<p>Get trigger delay in samples.</p>	<p>rp_AcqGetTriggerDelay</p>
<p>ACQ:TRIG:DLY:NS <par></p> <p>Example: ACQ:TRIG:DLY:NS 128</p>	<p><par>={value in ns}</p> <p>Default: 0</p>	<p>Set trigger delay in ns.</p>	<p>rp_AcqSetTriggerDelayNs</p>
<p>ACQ:TRIG:DLY:NS?</p> <p>Example: ACQ:TRIG:DLY:NS?</p> <p>Query return: 128 ns</p>		<p>Get trigger delay in ns.</p>	<p>rp_AcqGetTriggerDelayNs</p>
<p>ACQ:SOUR<n>:GAIN <par></p> <p>Example: ACQ:SOUR1:GAIN LV</p>	<p><par>={LV,HV}</p> <p>Default: LV</p>	<p>Set gain settings to HIGH or LOW. This gain is referring to jumper settings on Red Pitaya fast analog inputs.</p>	<p>rp_AcqSetGain</p>
<p>ACQ:TRIG:LEV <par></p> <p>Example: ACQ:TRIG:LEV 125 mV</p>	<p><par>={value in mV}</p> <p>Default: 0</p>	<p>Set trigger level in mV.</p>	<p>rp_AcqSetChannelThreshold</p>
<p>ACQ:TRIG:LEV?</p> <p>Example: ACQ:TRIG:LEV?</p> <p>Query return: 123 mV</p>		<p>Get trigger level in mV.</p>	<p>rp_AcqGetChannelThreshold)</p>

Data pointers			
ACQ:WPOS? Example: ACQ:WPOS? Query return: {write pointer position}		Returns current position of write pointer.	rp_AcqGetWritePointer
ACQ:TPOS? Example: ACQ:TPOS? Query return: 1234		Returns position where trigger event appeared.	rp_AcqGetWritePointerAtTrigger
Data read			
ACQ:DATA:UNITS <PAR> Example: ACQ:GET:DATA:UNITS RAW	<par>={RAW, VOLTS} Default: VOLTS	Selects units in which acquired data will be returned.	rp_AcqScpiDataUnits
ACQ:DATA:FORMAT <PAR> Example: ACQ:GET:DATA:FORMAT ASCII	<par>={FLOAT, ASCII} Default: FLOAT	Selects format acquired data will be returned.	rp_AcqScpiDataFormat
ACQ:SOUR<n>:DATA:STA:END? <start_pos>,<end_pos> Example: ACQ:SOUR1:GET:DATA 10,13 Query return: {123,231,-231}	<start_pos>={0,1,...,16384} <stop_pos>={0,1,...,16384} stop_pos > start_pos	Read samples from start to stop position.	rp_AcqGetDataPosRaw rp_AcqGetDataPosV
ACQ:SOUR<n>:DATA:STA:N? <start_pos>,<m> Example: ACQ:SOUR1:DATA? 10,3 Query return: {1.2,3.2,-1.2}		Read m samples from start position on.	rp_AcqGetDataRaw rp_AcqGetDataV
ACQ:SOUR<n>:DATA? Example: ACQ:SOUR2:DATA?		Read full buf. size starting from oldest sample in buffer (this is first sample after trigger delay). Trigger	rp_AcqGetOldestDataRaw rp_AcqGetOldestDataV size=buf_size !

<p>Query return: {1.2, 3.2, ..., -1.2}</p>		<p>delay by default is set to zero (in samples or in seconds). If trigger delay is set to zero it will read full buf. size starting from trigger.</p>	
<p>ACQ: SOUR<n>: DATA: OLD: N? <m></p> <p>Example: ACQ: SOUR2: DATA: OLD? 3</p> <p>Query return: {1.2, 3.2, -1.2}</p>		<p>Read m samples after trigger delay, starting from oldest sample in buffer (this is first sample after trigger delay). Trigger delay by default is set to zero (in samples or in seconds). If trigger delay is set to zero it will read m samples starting from trigger.</p>	<p>rp_AcqGetOldestDataRaw rp_AcqGetOldestDataV</p>
<p>ACQ: SOUR<n>: DATA: LAT: N? <m></p> <p>Example: ACQ: SOUR1: DATA: LAT? 3</p> <p>Query return: {1.2, 3.2, -1.2}</p>		<p>Read m samples before trigger delay. Trigger delay by default is set to zero (in samples or in seconds). If trigger delay is set to zero it will read m samples before trigger</p>	<p>rp_AcqGetLatestDataRaw rp_AcqGetLatestDataV</p>



<p>ACQ: BUF: SIZE?</p> <p>Example: ACQ: BUF: SIZE?</p> <p>Query return: 16384</p>		<p>Returns buffer size.</p>	<p>rp_AcqGetBufSize</p>
--	--	-----------------------------	--------------------------------

