QICK Time Tagger

Introduction

A time tagger, also known as a time-to-digital converter, is a device that measures the precise time an analog input crosses a specific voltage threshold. Time tagging measurements can determine the exact time of electrical events with picosecond precision. Additionally, it can be used to calculate the time-of-flight by measuring the difference between start and stop signals, creating a time tag upon photon detection.

Features

The Qick Time Tagger (QTT) block is designed to process ADC input signals for precise time tagging with a range of preprocessing options. This block is particularly useful for applications involving Superconducting Nanowire Single-Photon Detectors (SNSPDs) where signal characteristics might vary.

Below are the key features and functionalities of the Time Tagger block:

Signal Preprocessing:

- Signal Inversion: The input signal can be inverted to accommodate SNSPDs that have a negative voltage output. This ensures compatibility with various signal polarities.
- Filtering: The block provides an option to filter the input signal by averaging the current sample with the previous sample. This helps in smoothing out noise and fluctuations in the signal, leading to more stable and accurate time tagging.

Signal Selection:

- Sample Value: The actual sample value of the input signal can be used for comparison with the threshold.
- Slope Value: The first derivative (Current Sample – Previous Sample) of the input signal can be computed and used for comparison. This is particularly useful for detecting changes in the signal more sensitively.

Threshold Comparison:

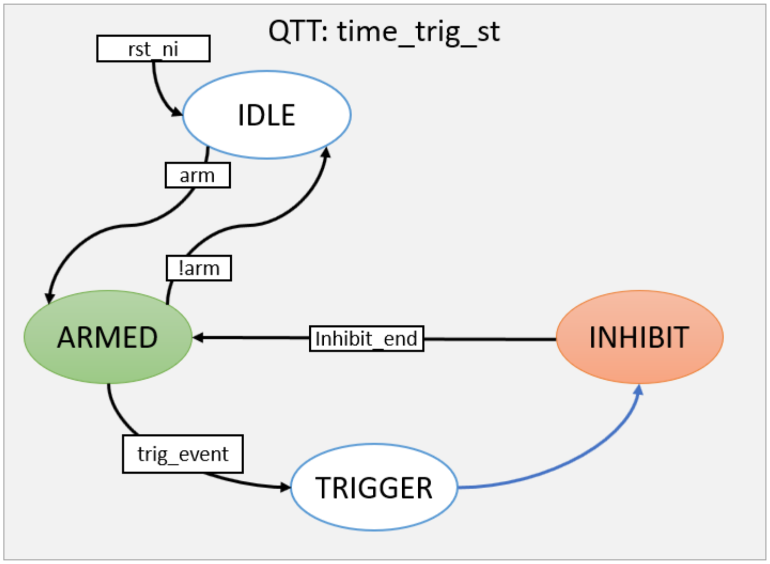
- The preprocessed signal (either the sample value or the slope value) is compared with a predefined threshold.

Interpolation for Precision:

- After the threshold comparison, the block can perform interpolation to enhance the precision of the time tag. This feature allows for more accurate determination of the crossing time, which is critical in high-precision applications.

Operation Overview:

- **Arming the Module**: The Time Tagger module is armed using the ARM command, which can be issued by a Python script, the tProc, or an external signal.
- **Monitoring and Comparison**: Once armed, the system continuously monitors the ADC input signals (up to four inputs) and compares each with its respective set threshold.
- **Trigger Event and Time Tag Generation**: If any input signal exceeds its threshold, the module generates a trigger event for that input, records a time tag, increments the ARM tag counter by one and initiates the process of saving ADC samples, storing up to 256 samples.
- **Inhibition State**: After generating a trigger event, the system enters an inhibition state. During this state, it waits for a specified number of pulses, as defined by the inhibit_pulses configuration value.
- **Return to Armed State**: Upon completing the inhibition period, the system returns to the armed state, ready to detect and process the next event.

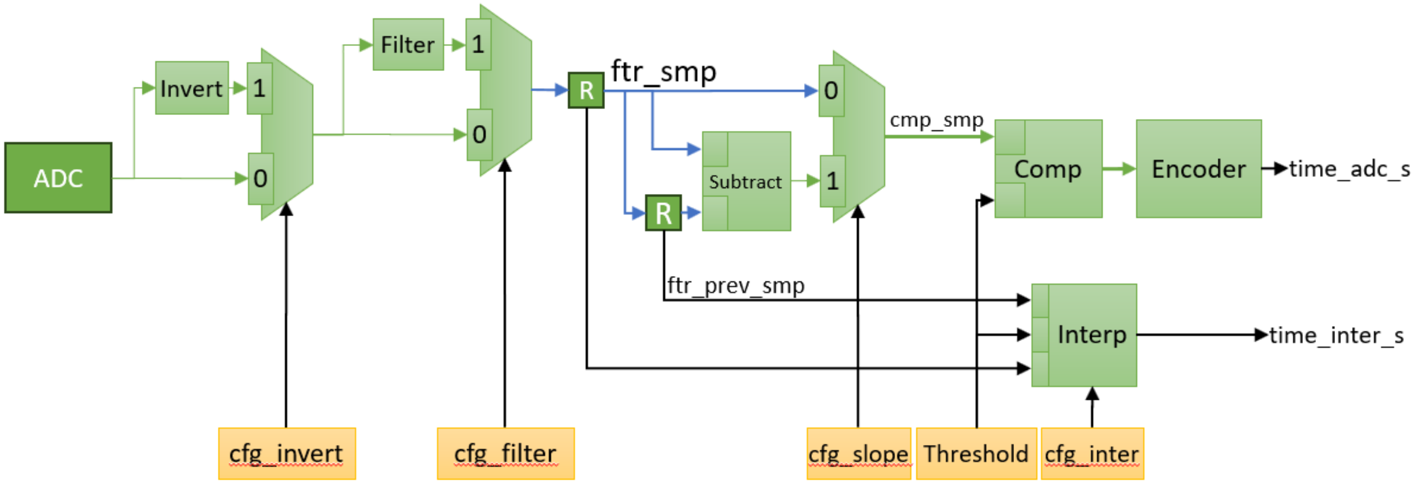


Python Driver

```
arm(cfg_filter, cfg_slope, cfg_inter, smp_wr_qty=0, cfg_invert=0)
disarm()
pop_dt
set_threshold(Value)
set_dead_time(Clocks)
read_mem(mem_sel:str, length=-1)

info()
print_axi_regs()
print_status()
print_debug()
```

Configuration Values



gfg_invert : An optional feature that inverts the input signal if configured. This is controlled by a signal inversion enable input.

cfg_filter: A moving average filter that averages the current sample with the previous sample. This can be enabled or disabled based on the configuration.

cfg_slope: The block calculates the first derivative of the input signal to provide an option for using the slope in threshold comparison.

Threshold : Compares the selected signal (either the filtered sample value or the derivative) against a predefined threshold value to determine the threshold crossing event.

cfg_inter: Enhances the precision of the time tag by interpolating the exact crossing point, providing finer resolution in time tagging.

Arm

The arm I/O signal

Outputs>

The difference between trig_o and cmp_o

Cmp_o is the output of the comparator

Trig_o is the output of a triggered signal, it happens if the system is not inhibited and after the intyerpolation is done.

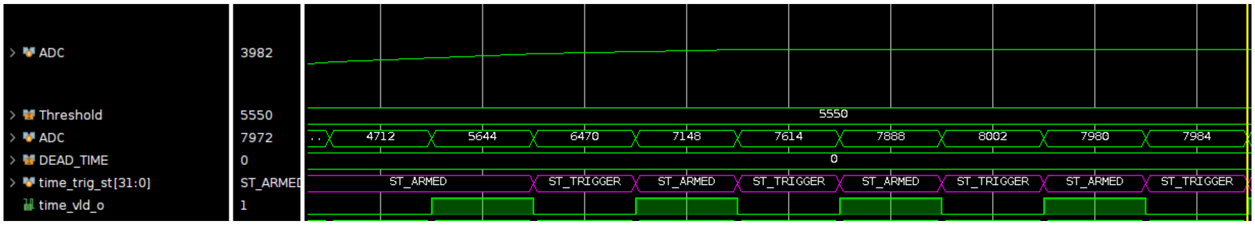
I counts the number of tags

Synthesis Utilization

TYPE	ADC_QTY	INTER	ARM	SMP	TAG_AW	ARM_AW	SMP_AW	LUT – Log	LUT - DFF	CARRY8	F7MUX	BRAM
DEFAULT	4	7	1	1	19	19	19					
DEFAULT	4	4	1	1	16	10	19					
DEFAULT	4	4	1	1	16	10	16					
DEFAULT	4	4	1	0	16	10	0					
DEFAULT	1	4	1	0	20	10						
DEFAULT	1	4	1	0	19	10						
DEFAULT	1	4	1	0	18	10						
DEFAULT	1	4	1	0	16	10						
DEFAULT	1	4	1	0	16	10	18	1738	1219	98	24	60
							DSP	1710	1217			

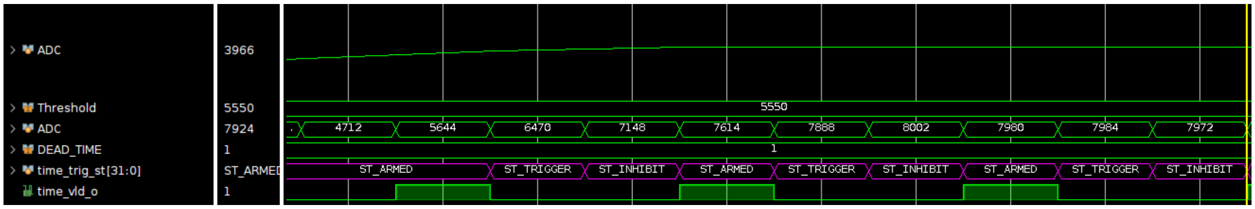
The "qick_time_tagger" module is a custom peripheral designed for time-tagging events based on a threshold.

DEAD TIME = 0

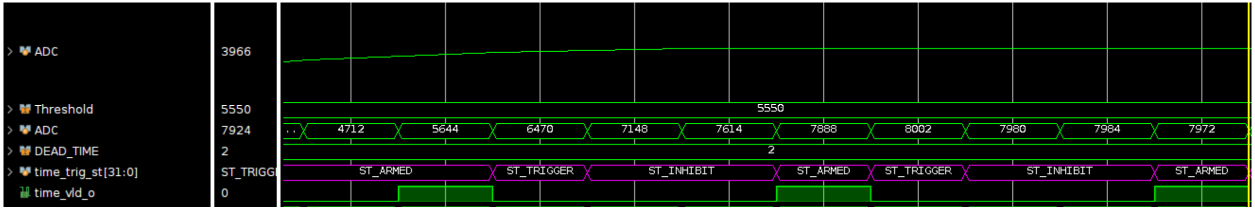


ADC 3982
Threshold 5550
ADC 7972
DEAD_TIME 0
time_trig_st[31:0] ST_ARMED
time_vld_o 1

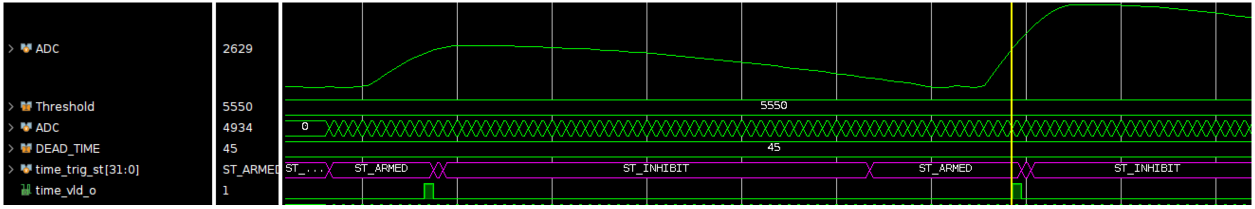
DEAD TIME = 1



DEAD_TIME = 2



DEAD_TIME = 40



It features a control state machine, a time counter, an inhibit counter, and a threshold comparator.

Operation Overview:

- The module is armed when the 'arm_i' input is asserted, transitioning from the IDLE state to ARMED state.
- Upon arming, the system monitors the input 'adc_dt' and compares it with the threshold 'cmp_th_i'.
- If 'adc_dt' exceeds the threshold, it generates a trigger event and a time tag.
- The system then enters an inhibition state, waiting for a specified number of pulses (defined by the input 'cmp_inh_i').
- After inhibition, the system returns to the armed state for the next event.

Module Inputs:

- clk_i: Clock input
- rst_ni: Active-low asynchronous reset
- arm_i: Arm trigger input
- cmp_th_i: Threshold data input for comparison
- cmp_inh_i: Inhibit pulses input
- adc_dt: Input data for comparison

Module Outputs:

- tag_wr_o: Output indicating when a time tag is generated
- tag_dt_o: Output providing the time tag information

State Machine:

- ST_IDLE: Initial state, waiting for the arm trigger.
- ST_ARMED: Armed state, monitoring 'adc_dt' for comparison events.
- ST_TRIGGER: State following a comparison event, preparing for inhibition.
- ST_INHIBIT: Inhibition state, waiting for a specified number of pulses.

Counters:

- time_cnt: Time counter, increments on each clock cycle during the armed state.
- inhibit_cnt: Inhibition counter, increments during the inhibition state.

Threshold Comparison:

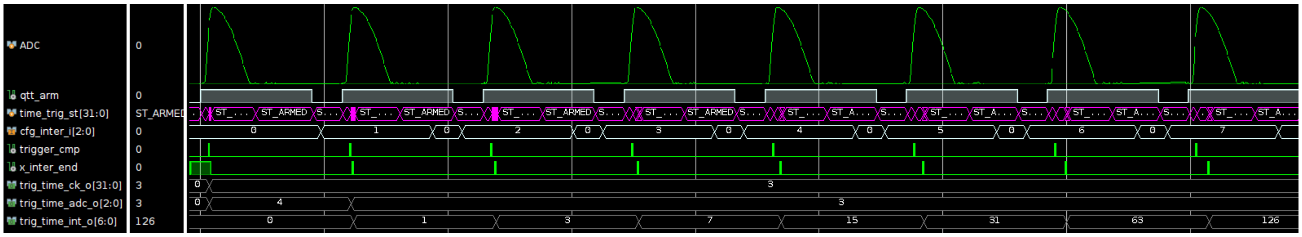
- The module uses a threshold comparator ('qick_adc_thr_cmp') to compare 'adc_dt' with 'cmp_th_i'.

Time Tag Generation:
- When a trigger event occurs, the module generates a time tag by combining 'time_cnt' and 'trig_inter'.

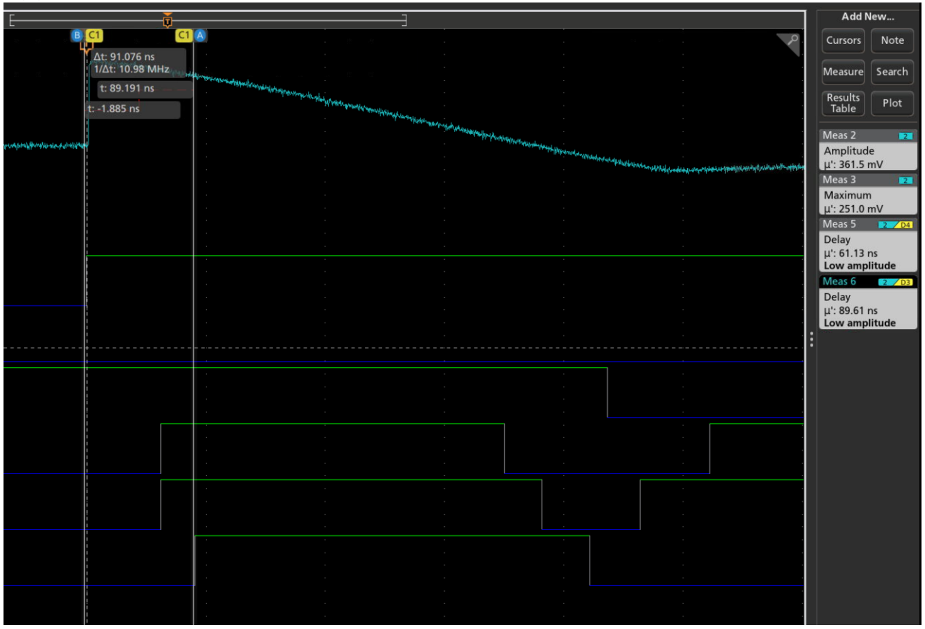
Optional

- MAX
- MIN
- Average (using Running Average) SUM / N
- MAD (Mean absolute Deviation)

Once FIFO is FULL…
The most old values will be rewritten (Lost).



Using the Time tagger as Comparator for Trigger
Appendix 1



Δt: 91.076 ns
1/Δt: 10.98 MHz
t: 89.191 ns
t: -1.885 ns

Meas 2 — 2
Amplitude
µ': 361.5 mV

Meas 3 — 2
Maximum
µ': 251.0 mV

Meas 5 — 2 / D4
Delay
µ': 61.13 ns
Low amplitude

Meas 6 — 2 / D3
Delay
µ': 89.61 ns
Low amplitude

Time comparator
With a 4Ghz Bandwith
The Internal Comparator of the ADC takes 60ns to compare, the qick time tagger takes 90ns
The blue is the Analog Input to the ADC
D3 is the QTT comparator Output
D4 is the ADC internal Threshold