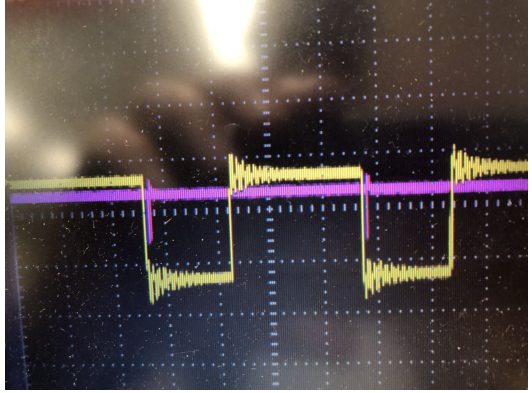
HDSoC Deadtime Studies

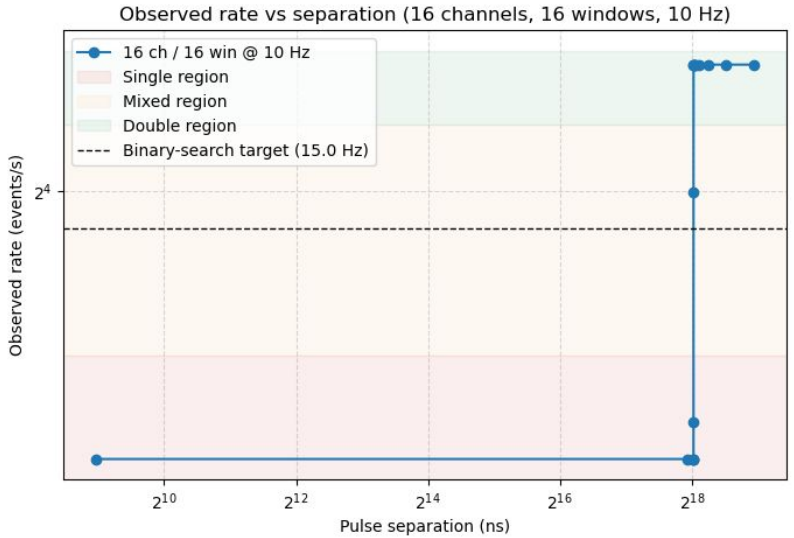
Jack Carlton
University of Kentucky

HDSoC Deadtime (Methods)

- Used Raspberry Pi Pico W + NIM modules to generated configurable double pulse signal
- Used HDSoC DAQ to observe event rate for varying parameters
- See spike corresponding to deadtime



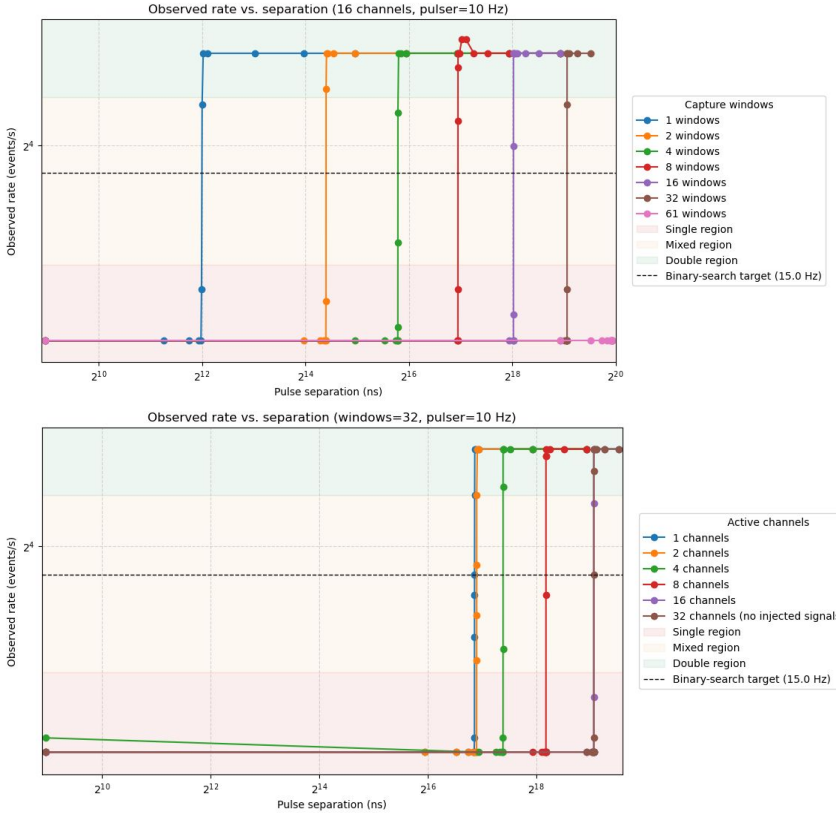
Example Double pulse on Oscilloscope



Example Rate Response to Double Pulse Separation

HDSoC Deadtime (Parameter Scan)

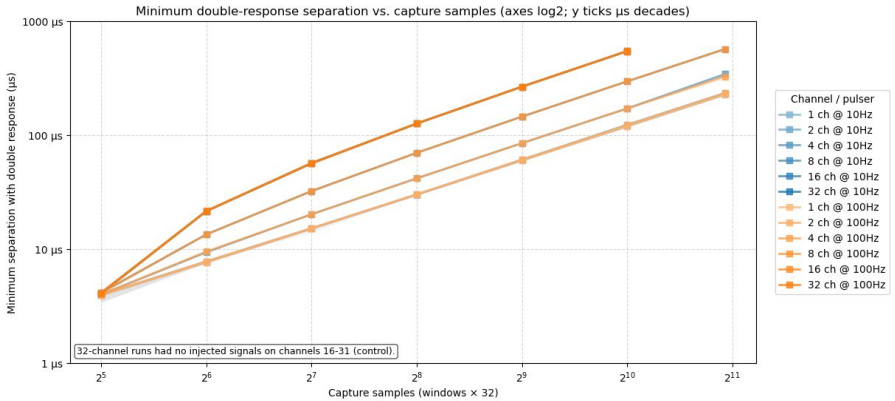
- The deadtime depends on:
  - Digitization window length
  - Number of channels actively receiving pulses
- It does not depend on:
  - Number of inactive channels
  - Rate of events (up to some threshold, likely corresponding to ~55 MB/s)
- We tested the parameter space defined by
  - Windows = [1,2,4,8,16,32,61]
  - Channels = [1,2,4,8,16,32]
  - Input Signal Rate = [10 Hz, 100 Hz]
- Space defined as:
  - Windows ⊗ Channels ⊗ Input Signal Rate
- Note:
  - Signals were fanned out to channels 0-15, no signals were in channels 16-31

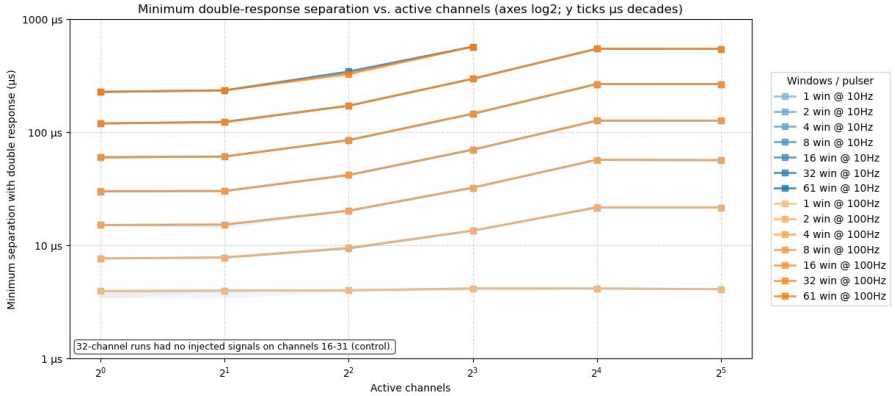


Examples showing channels and windows affect on deadtime

HDSoC Deadtime (Results)

- Deadtime increases (~linearly) with
  - # of active channels
  - # of windows (# of samples, 1 window == 32 samples, this is nalu's terminology)
- Deadtime unaffected by:
  - Rate
    - 10Hz and 100Hz input signal curves overlap
  - Inactive channels
    - Curve flattens when channels 16-31 (no input) are enabled in software
- [JSON file of deadtime upper bounds available]





Auxiliary Slides

More information

- More plots in jupyter notebook (can view on github)
  - https://github.com/jaca230/nalu_deadtime_tests/blob/main/notebooks/deadtime_vs_separation.ipynb
- More information documented in Josh's demonstration simulation upgrade google doc
  - https://docs.google.com/document/d/18LLyw50G9MPmdMN1EXKJsDdKYHJieBN8n_xYpL8UGR0/edit?pli=1&tab=t.0