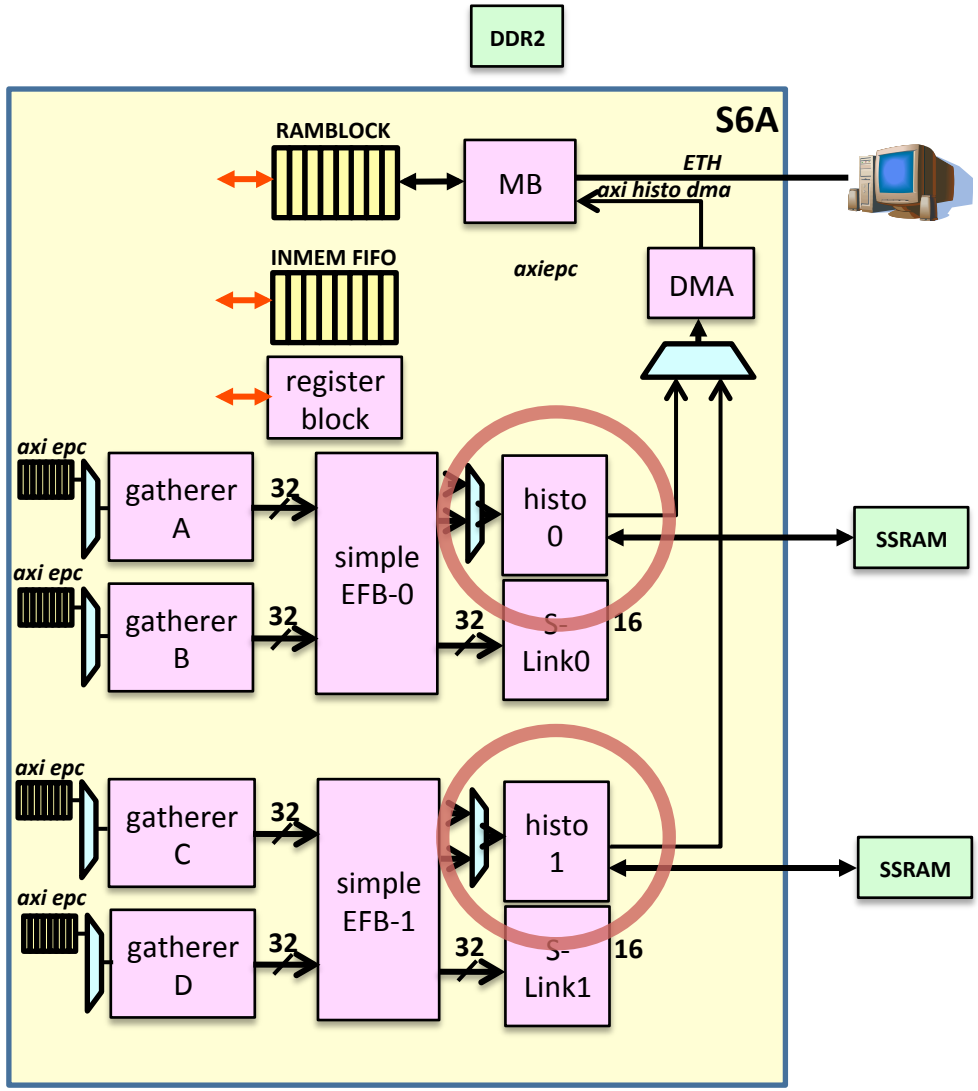
“New” Histogrammer Summary

moritz.kretz@cern.ch

What does the Histogrammer do?

- block on the ROD slave FPGA that is fed per pixel data from the EFB
- computes and stores occupancy, ∑ ToT, ∑ ToT^2
- stores data internally (now: FPGA BRAM, soonish: external SSRAM)
- is steered by the MicroBlaze CPU on the FPGA
- readout of data from internal memory to DDR-RAM, which is accessible by the MicroBlaze (-> network transfer)
- two histogramming units per slave (8 FE-I4s/unit)



Changes

- uses 36 bits/pixel internally (instead of 32 bits)

(might again be cumbersome for synthesis, but we'll see...)

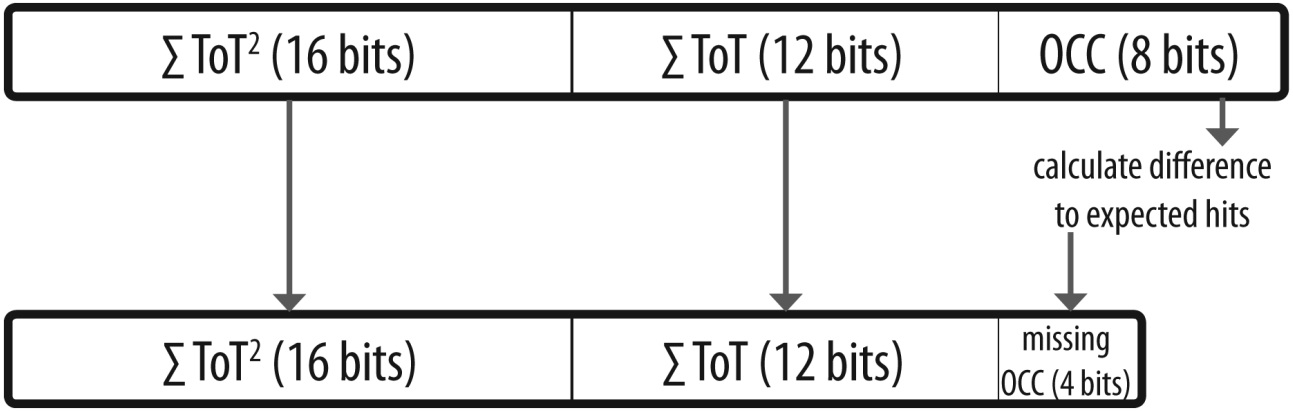
- new mode of operation for histogramming (only) occupancy during data-taking:
  24 bits -> maximum of 16M hits/pixel
- new readout modes (4 in total) and hence changes for the control register

Operation Modes

- 4 modes of operation:
  - SHORT_TOT
  - LONG_TOT
  - ONLINE_OCCUPANCY
  - OFFLINE_OCCUPANCY
- these should cover all foreseen use-cases of the histogrammer

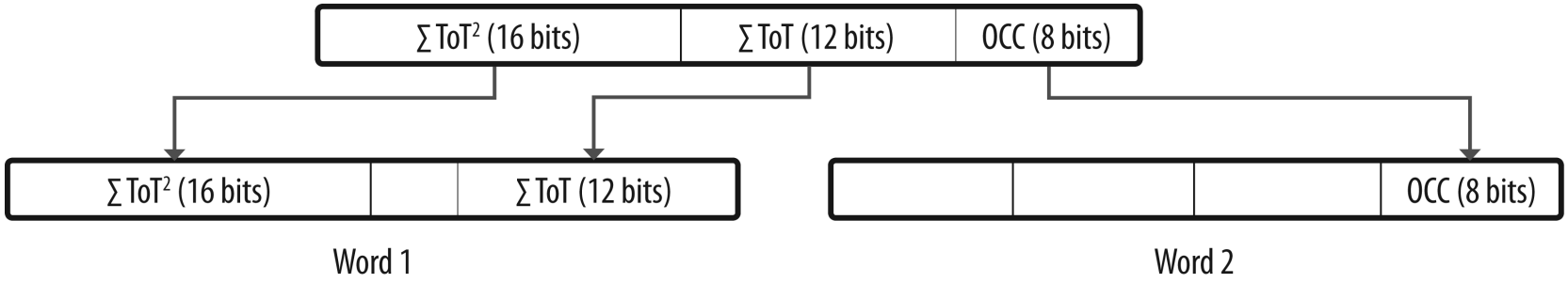
SHORT_TOT

- condense the 36 bit wide data to 32 bit by calculating the difference of actual triggers vs. expected triggers (0-15)
- used for scans that are well above threshold anyways



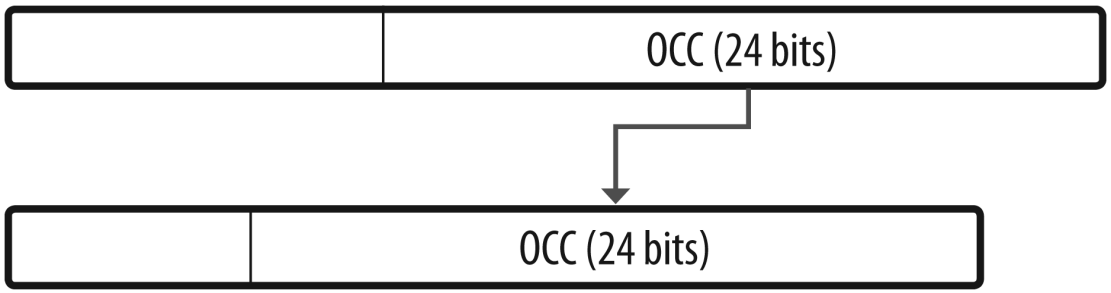
LONG_TOT

- output all of the internal data to two 32 bit wide words
- aligning word 1 fields on 16 bit border



ONLINE_OCCUPANCY

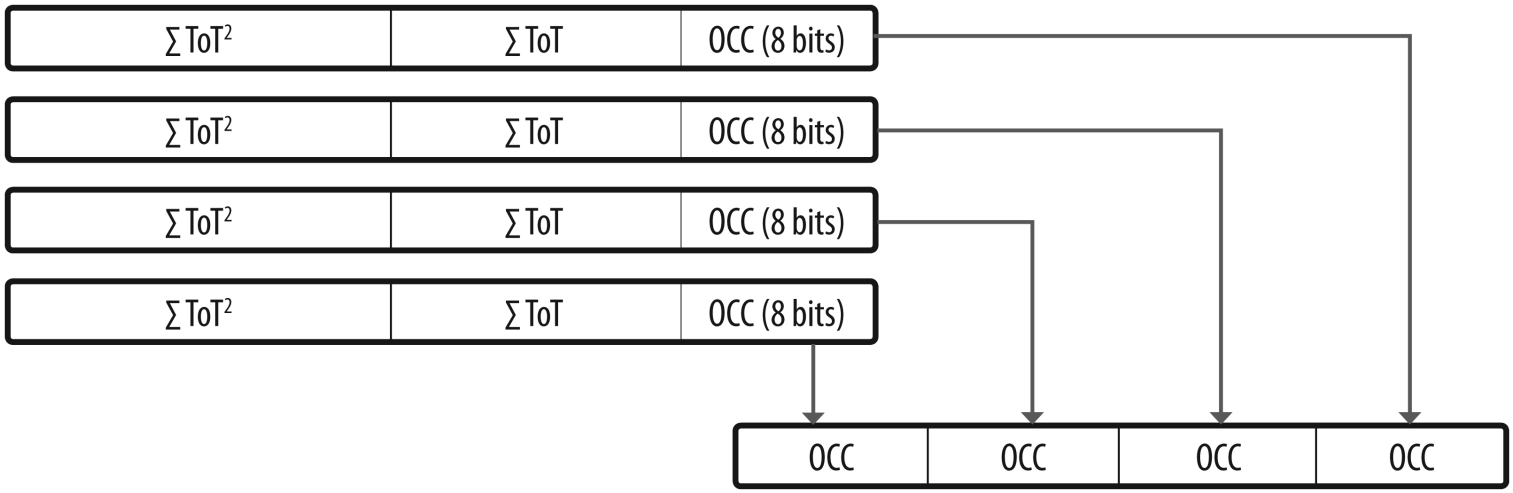
- internally only sample occupancy with 24 bit accuracy
- used during data taking



OFFLINE_OCCUPANCY

- throw away everything but occupancy and pack

  4 pixels in one 32 bit word



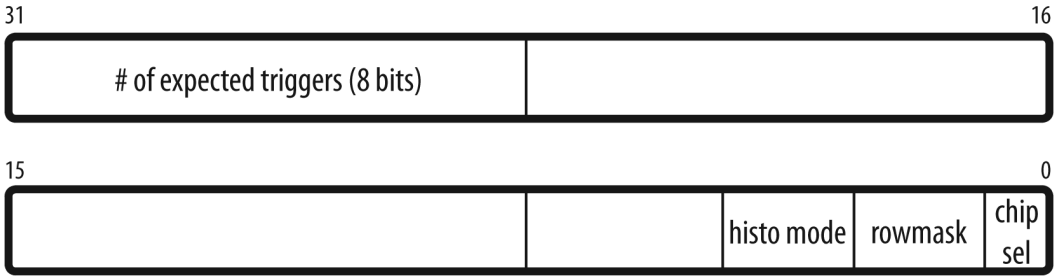
Mask Stepping et al.

- In some/most cases we don't sample all pixels, just a subset.

  Either...
  - **one** frontend or **all** frontends and
  - every row, every 2^{nd}, 4^{th}, 8^{th} row

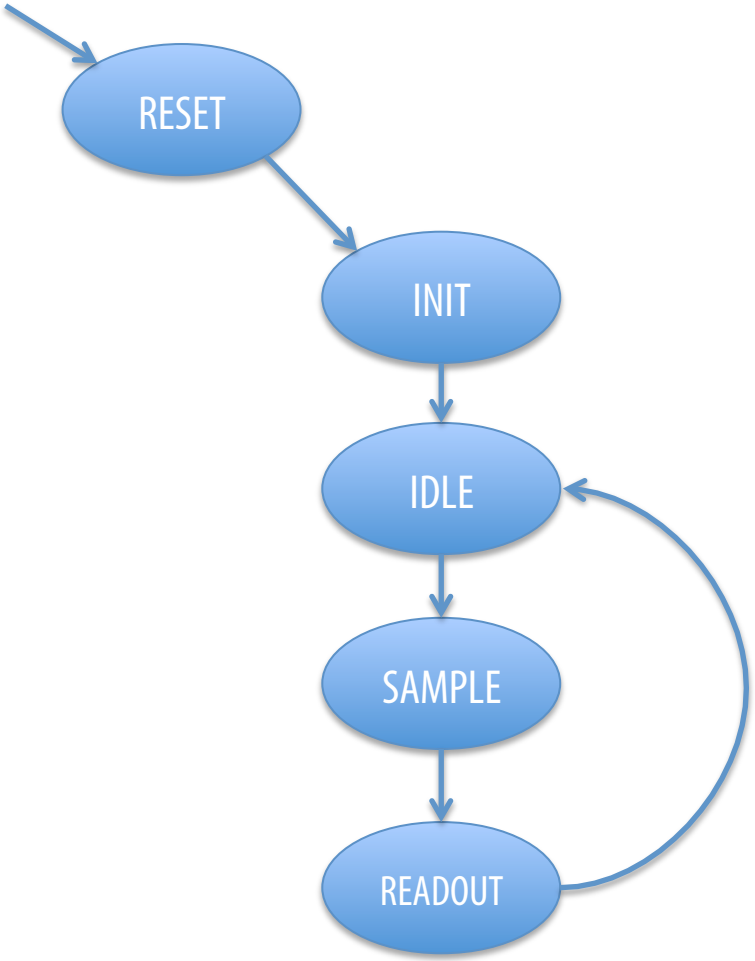
Control Register

- configuring of the histogrammer via the control register
- can be written/read to/from the MicroBlaze



Relevant States of Histogrammer

- **RESET**
- **INIT**: clean the internal memory
- **IDLE**: set configuration from control register
- **SAMPLE**: compute/store histogram data
- **READOUT**: reformat the data and move it to DDR-RAM via DMA transfer



Outlook

- ongoing debugging for LONG_TOT readout
- hopefully working implementation/bitfile ready within the next week
- more sophisticated testbench for the complete histogrammer (automated tests) – merge with Riccardo?
- error counters (invalid pixel locations, ...?)

Questions?