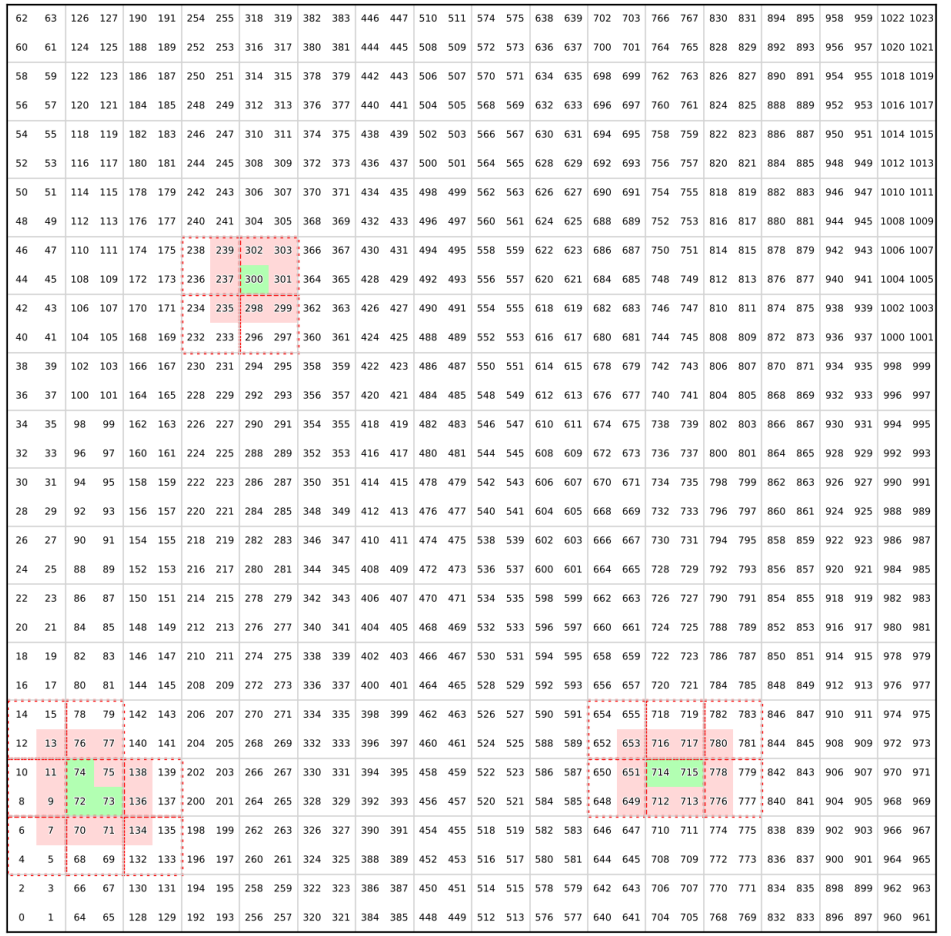
EICROC2 digital architecture

24 March 2026

Introduction

- AC LGAD sensors require to read the hit pixels but also their neighbors due to charge sharing

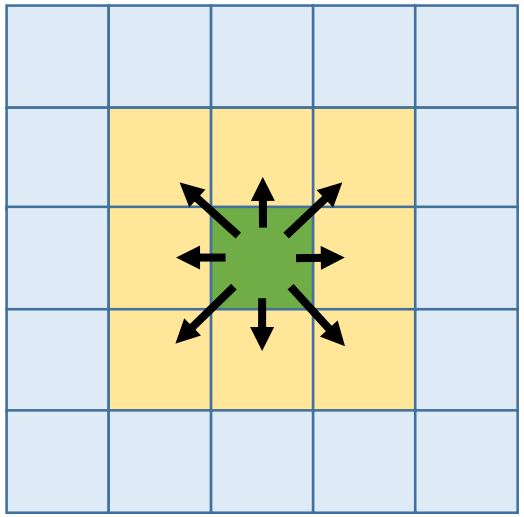
- EICROC2 digital readout global specifications:
  - Must read neighboring pixels
  - Small area (can use 130nm techno if it is small enough)
  - Low power (< 500µW/pixel)
  - Read fast enough to limit data loss



Triggers for neighboring pixels

TP : pixel receiving a hit, provide TDC and ADC data

TL1 : pixel receiving a trigger order from a TP, provide ADC data only



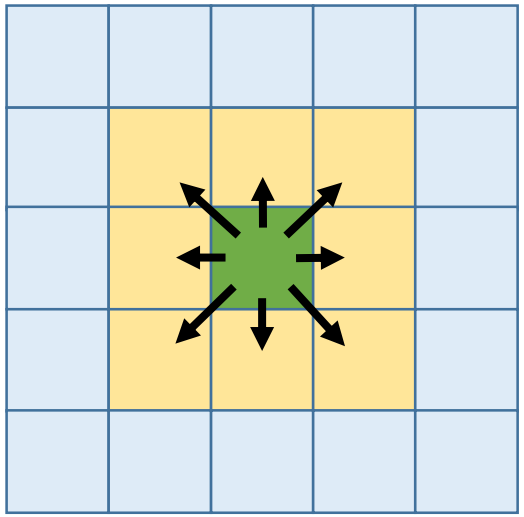
Case 1 : read 8 neighbors

Complex routing, hard to implement

Triggers for neighboring pixels

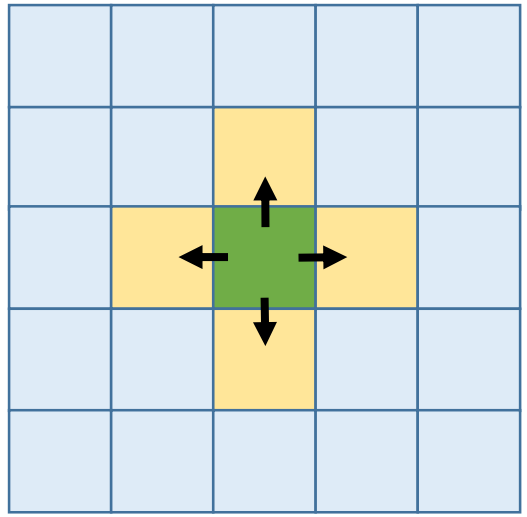
TP : pixel receiving a hit, provide TDC and ADC data

TL1 : pixel receiving a trigger order from a TP, provide ADC data only



Case 1 : read 8 neighbors

Complex routing, hard to implement



Case 2 : read 4 neighbors

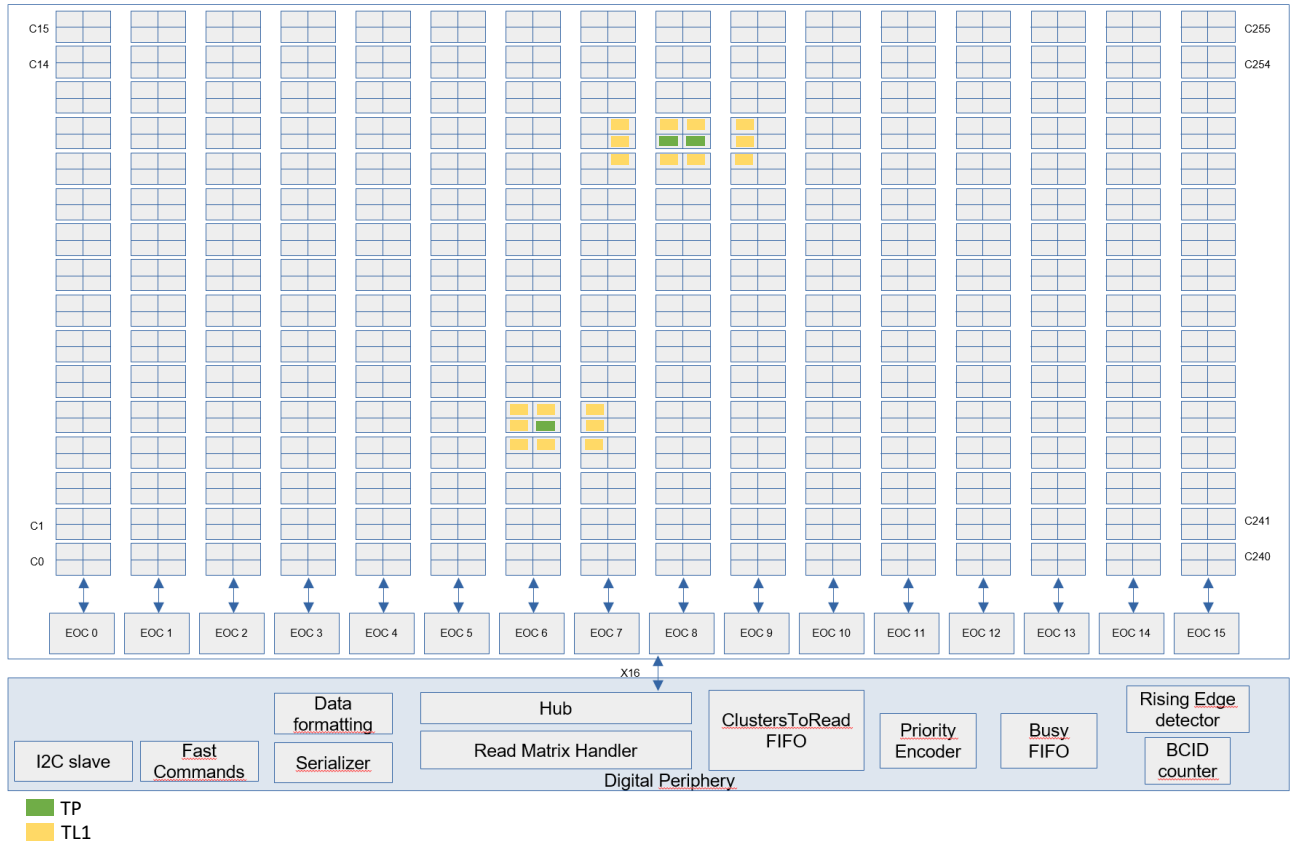
Automatic routing by abutting, easy to implement

- Trigger scheme is an important choice for the design, impact complexity of the chip
- Choice depend on sensors and how spatial resolution performance is impacted for each case

Readout architecture

- Goals
  - Minimize digital logic inside pixel/matrix
  - Minimize signals between matrix and periphery
  - Triplication of sensitive circuits
    - **No triplication on data path**
  - Complexity on periphery side, simple logic inside pixel

- Proposal : **Based on clusters of 4 pixels, controlled by the periphery. Data transmission using ready/valid protocol**
  - Some logic shared at cluster level
  - Shared data bus between 2 columns



First iteration of this architecture is ready, debug in progress

Dead time

What is the deadtime?

Dead time corresponds to the time a pixel is **blocked** after a hit and will not acquire new hits : pixel is **blind** during that time

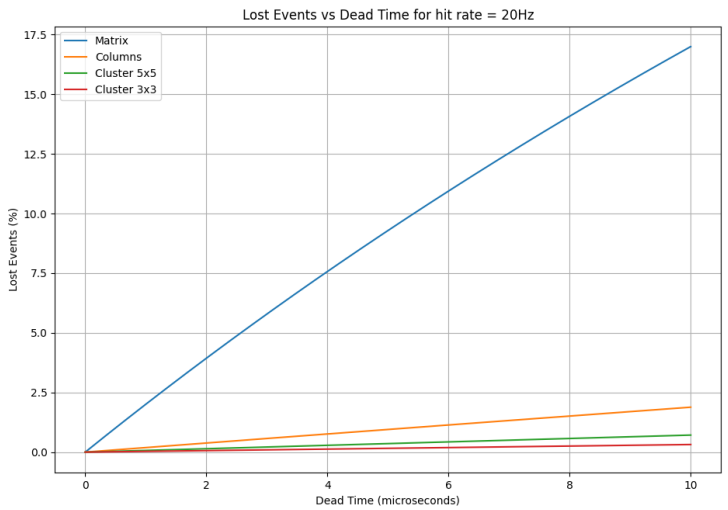
- Analog dead time corresponds to the **ADC conversion time**
- Digital dead time is the time to get the **data from the pixel to the periphery**

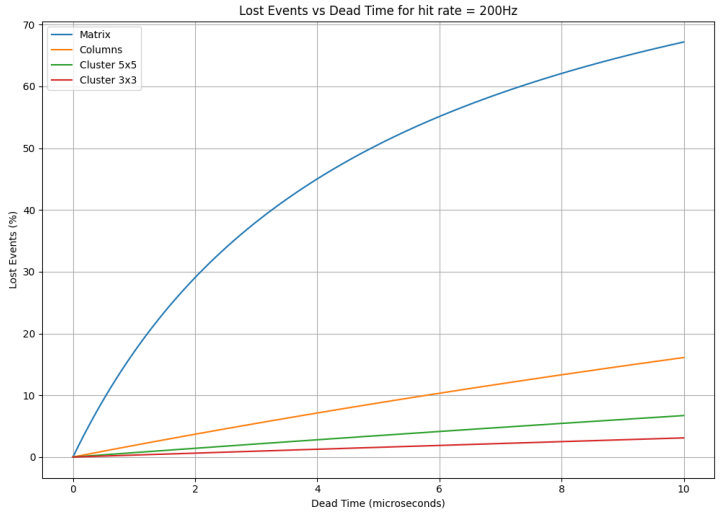
➔ Depending on the hit rate per pixel and this deadtime, we can have an estimation of the data loss for several cases

Data loss

- Data loss depend on how many pixels are blocked and for how long + pixel hit rate

- Plots show how dead time impacts data loss for different blocking cases





- **First estimation of data loss with proposed readout :** (considering average analog DeadTime=400ns)

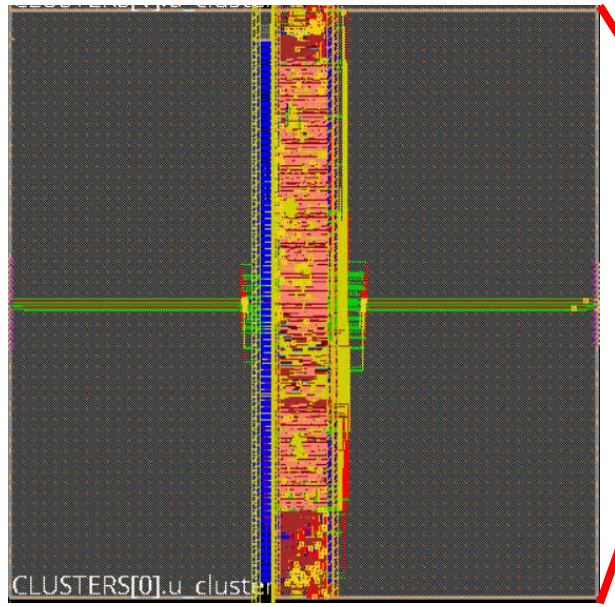
Data loss for Clusters, 3x3 readout	Rate = 20Hz	Rate = 200Hz
Default (1 hit = 1 TP, 8 TL1)	**0.028%**	**0.28%**
Multiple (2 hits = 2 TP, 10 TL1)	0.032%	0.32%
Worst case (4 hits = 4 TP, 12 TL1)	0.040%	0.40%

- Analyses can be redone with a more realistic hit rate per pixel

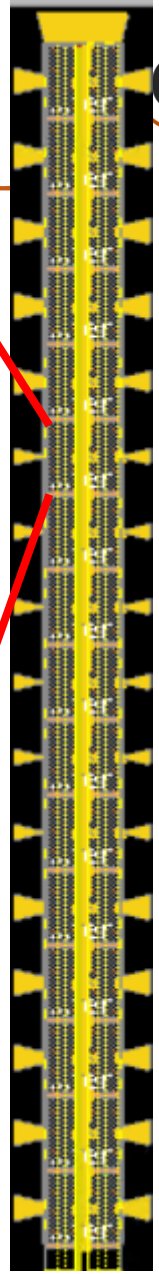
Physical implementation status

- First physical implementation done in **130nm**
  - Check if we manage to fit digital readout inside pixel
- Preliminary layout of the cluster of 4 pixels and the column
- Triplication of cluster (probably need some refinement)
- Using a pixel_analog_dummy macro, not the real analog front-end
- **Current architecture can fit using 130nm technology (barely)**

Improvement possible to reduce area if needed (configuration registers access buses, triplication?)



Cluster of 4 pixels



Column

Organisation and schedule

- LPCA team for EICROC :
  - 2 digital engineers (N. Kachkachi, A. Soulier) + 1 engineer part time on simulations (J. Bonnard)
  - 3 experts helping for specific points (H.Chanal for architecture, L.Royer for mixed-signal/system, N.Arveuf for UVM)
- Code and scripts shared on IN2P3 git
- Documentations shared on box IN2P3
  - EICROC spec : document gathering global specification related to the digital readout of the chip
  - Implementation spec : Detailed architecture explanation
- SOS server setup at Clermont : Centralized database for IP sharing

- Schedule planned:
  - First iteration of full digital RTL architecture including all blocks : Late Summer 2026
  - First iteration of full chip implementation : Late Autumn 2026

Backup

Triplication

- EICROC2 must be radiation hard, in particular tolerant to Single Event Effects (SEE)
  - SEE can cause bit-flips or transients on signal lines

  ➔ Make the design tolerant to SEE by triplicating sensitive circuits

- Significant impact on area and power consumption
- Dony by using TMRG tool (from CERN)

Simulations

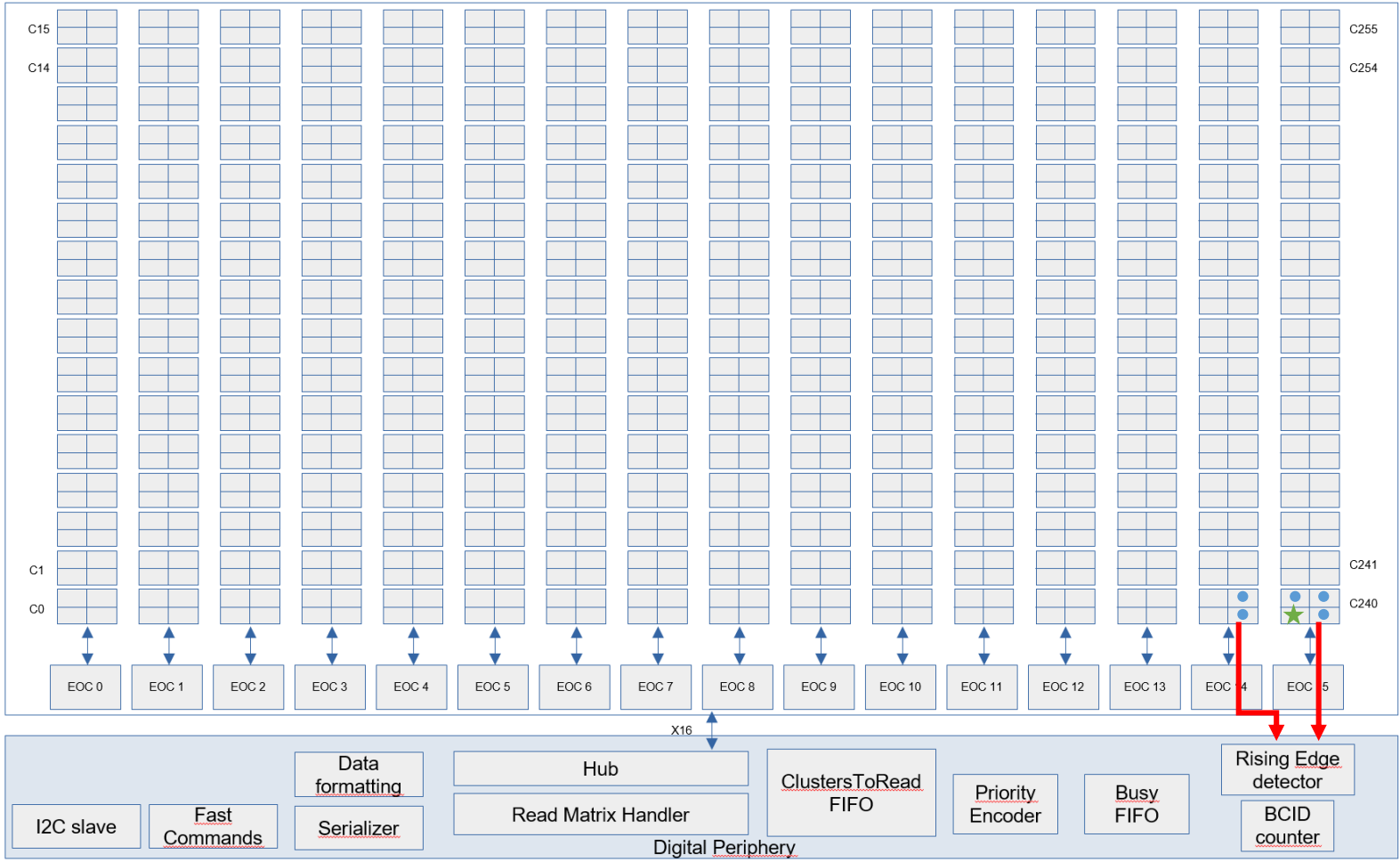
- Simulating 1024 pixels of the matrix

- Injecting in pixels, different tests already implemented :
  - One pixel receiving a hit
  - Side-by-side pixels receiving a hit
  - Pixel in a corner receiving a hit
  - Full matrix hit
  - Consecutive hits in same pixel

- A C++ model has been developped to do the following, using data output file from simulation :
  - Reproduce injection of TP pixel(s)
    - Check that injected pixels (TP+TL) are the same in model and in simulation
  - Compare number of pixel injected / number of pixel read
  - Compare ADC/TDC values for each pixel

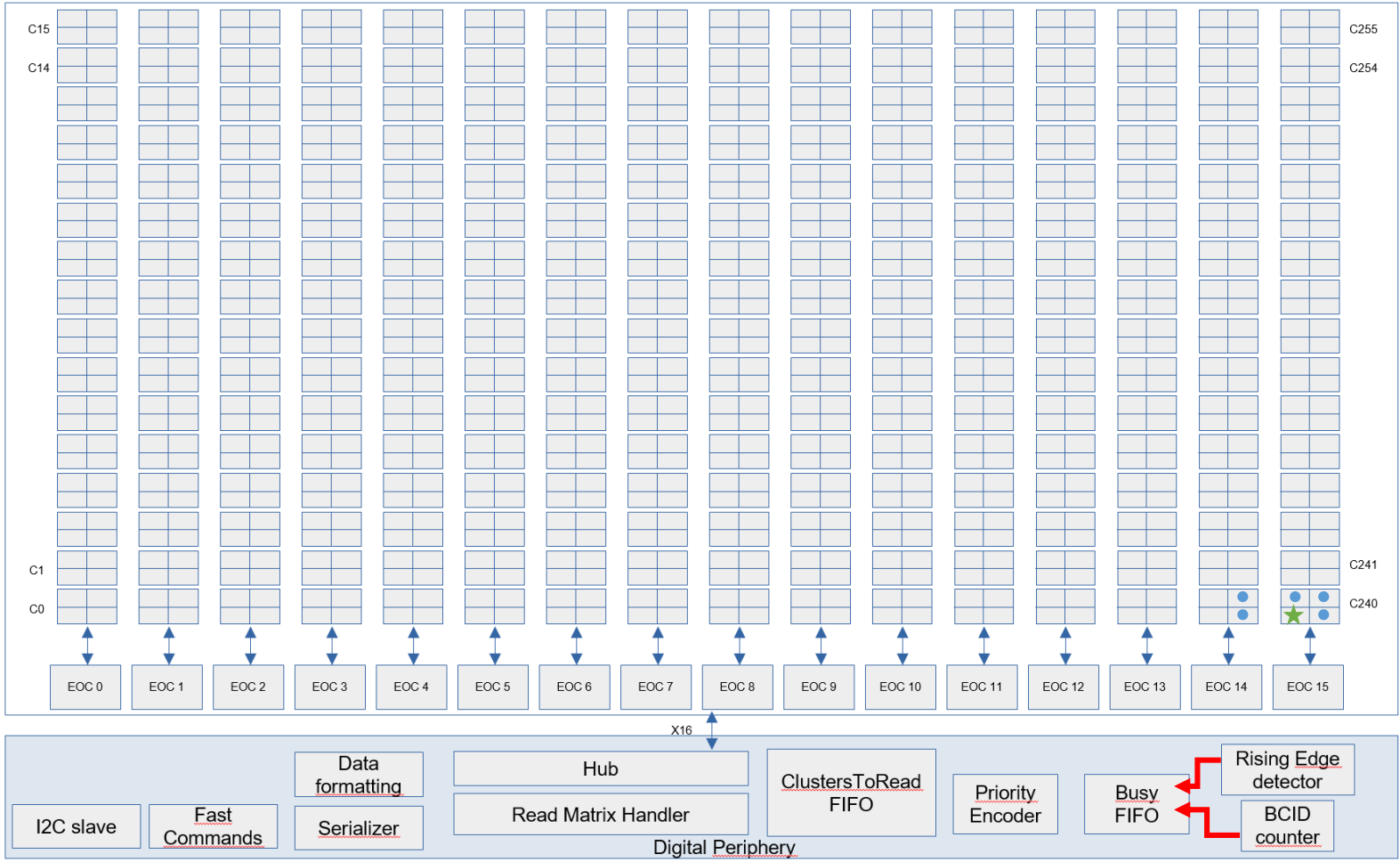
Architecture overview

1. Clusters send a busy signal to periphery when one of their pixels is hit



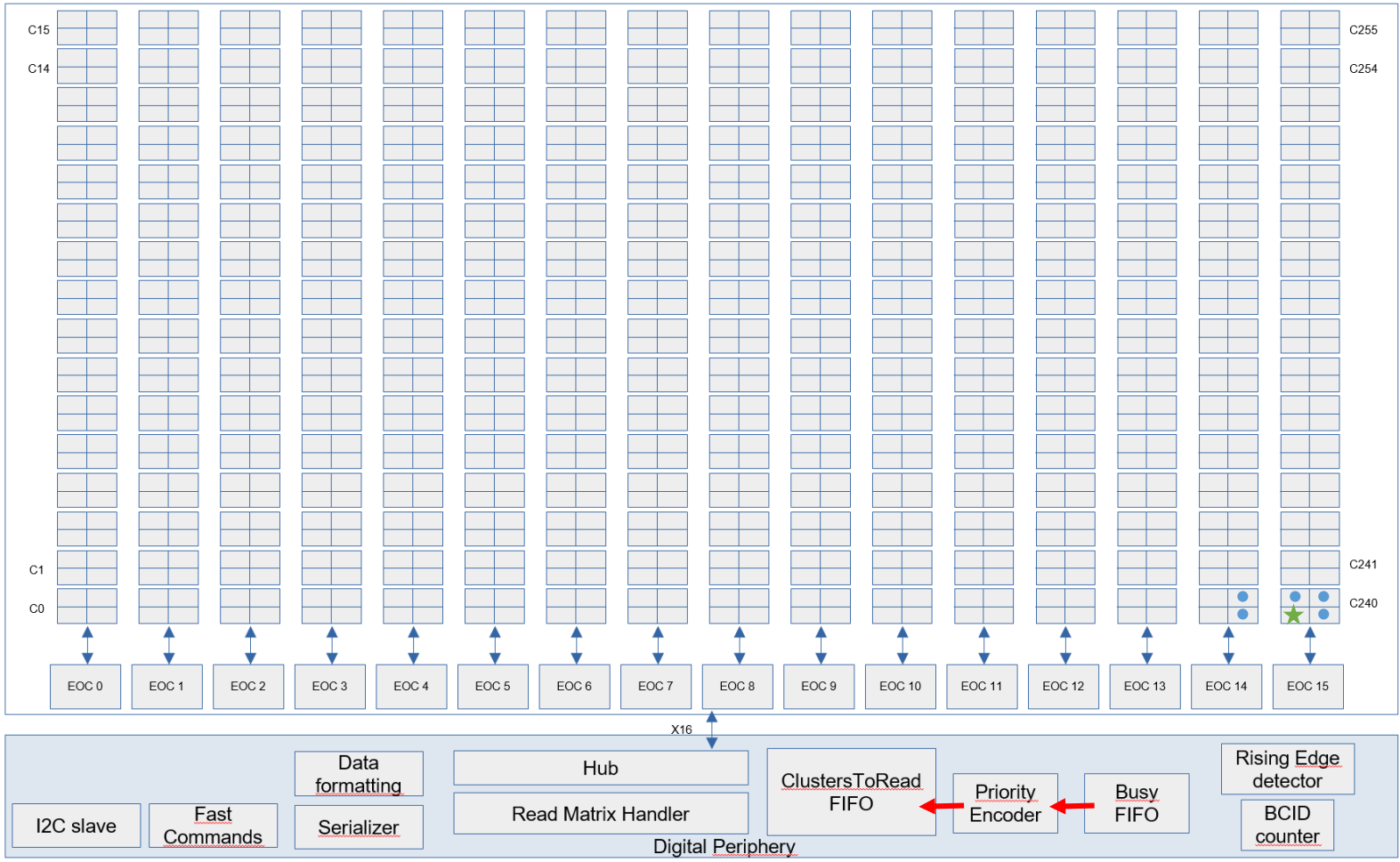
Architecture overview

1. Clusters send a busy signal to periphery when one of their pixels is hit
2. Periphery assign these signals to a timestamp (=BCID)



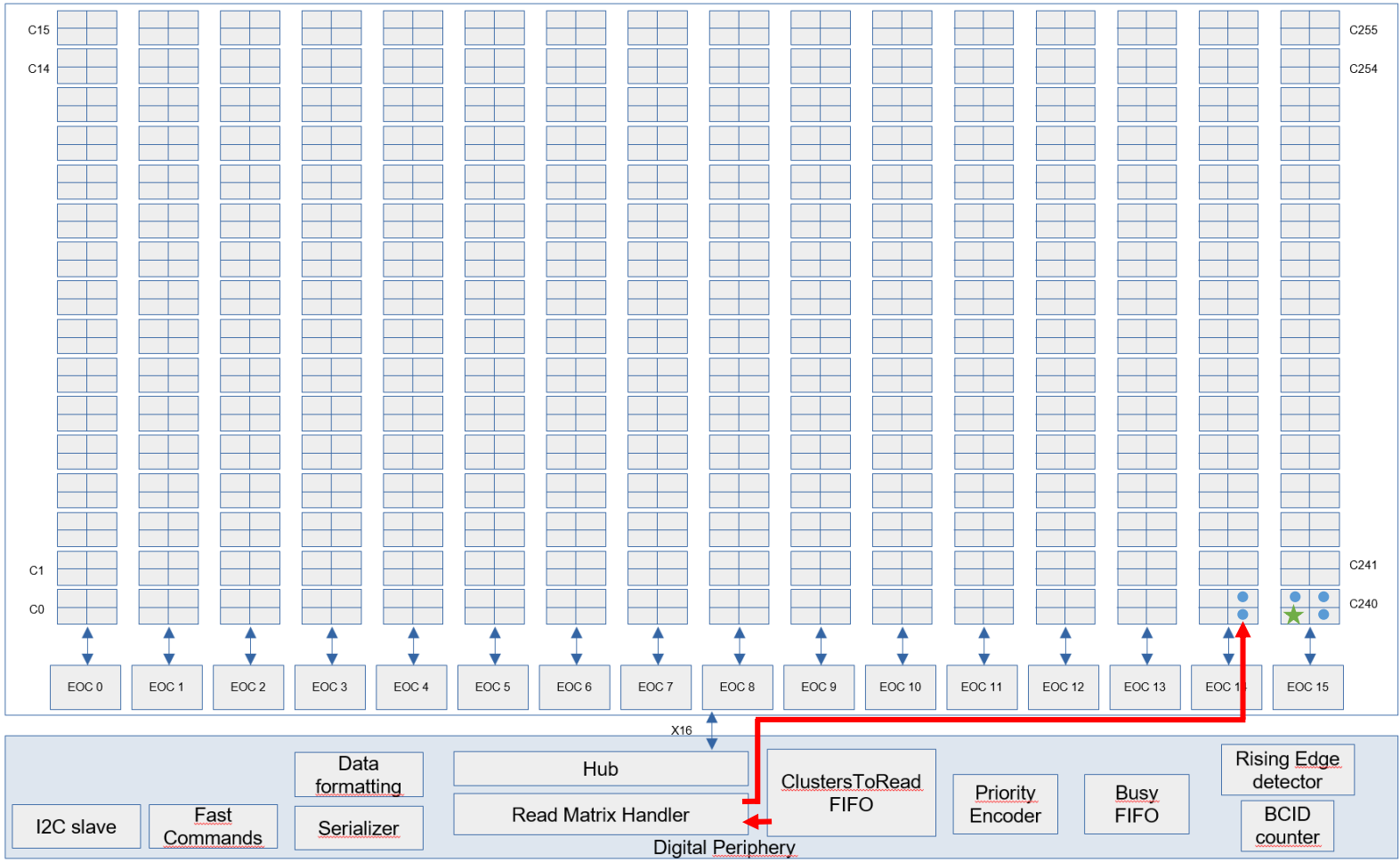
Architecture overview

1. Clusters send a busy signal to periphery when one of their pixels is hit
2. Periphery assign these signals to a timestamp (=BCID)
3. Addresses of clusters to read are retrieved from these signals and stored in a FIFO



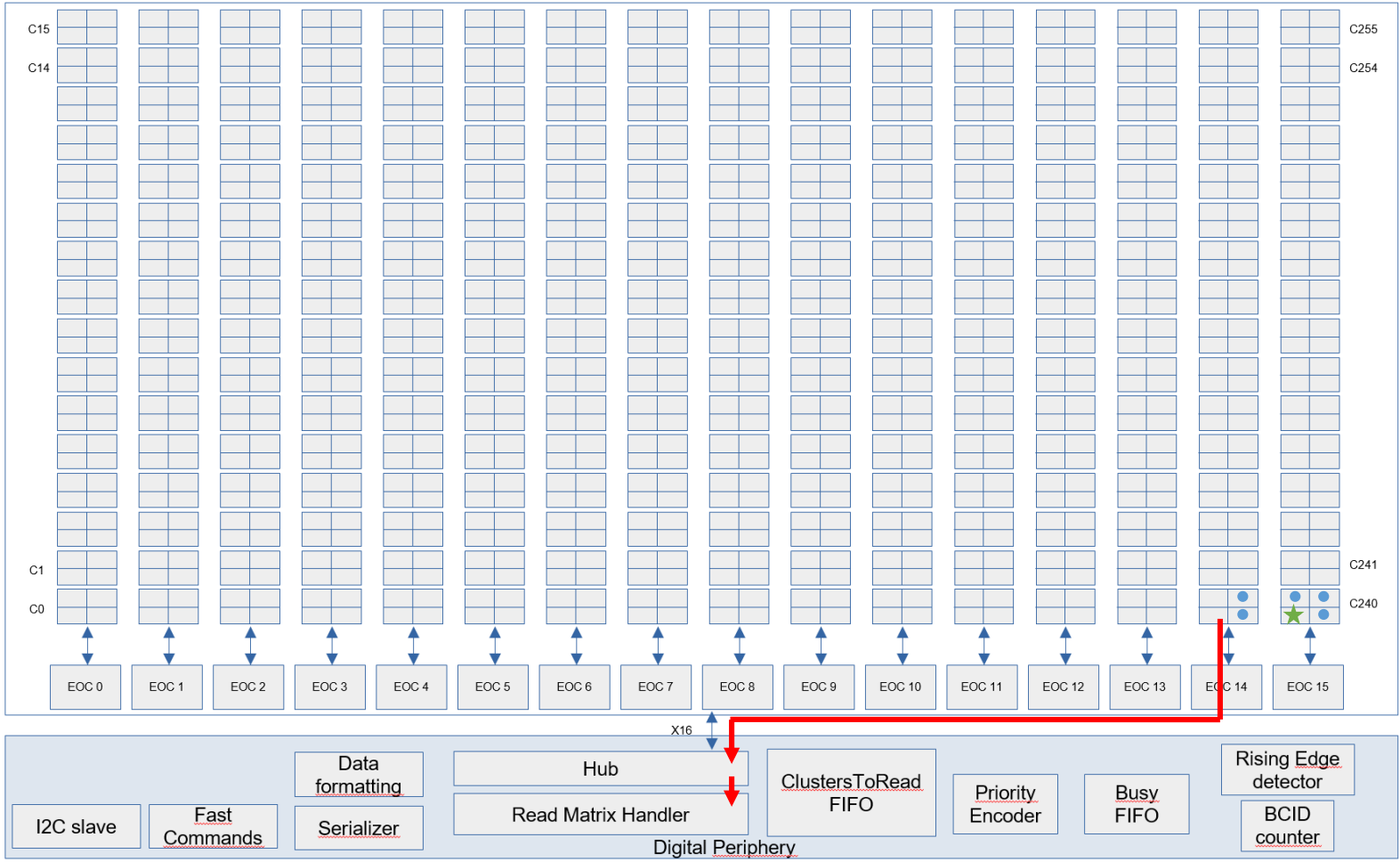
Architecture overview

1. Clusters send a busy signal to periphery when one of their pixels is hit
2. Periphery assign these signals to a timestamp (=BCID)
3. Addresses of clusters to read are retrieved from these signals and stored in a FIFO
4. Read requests are sent to relevant clusters



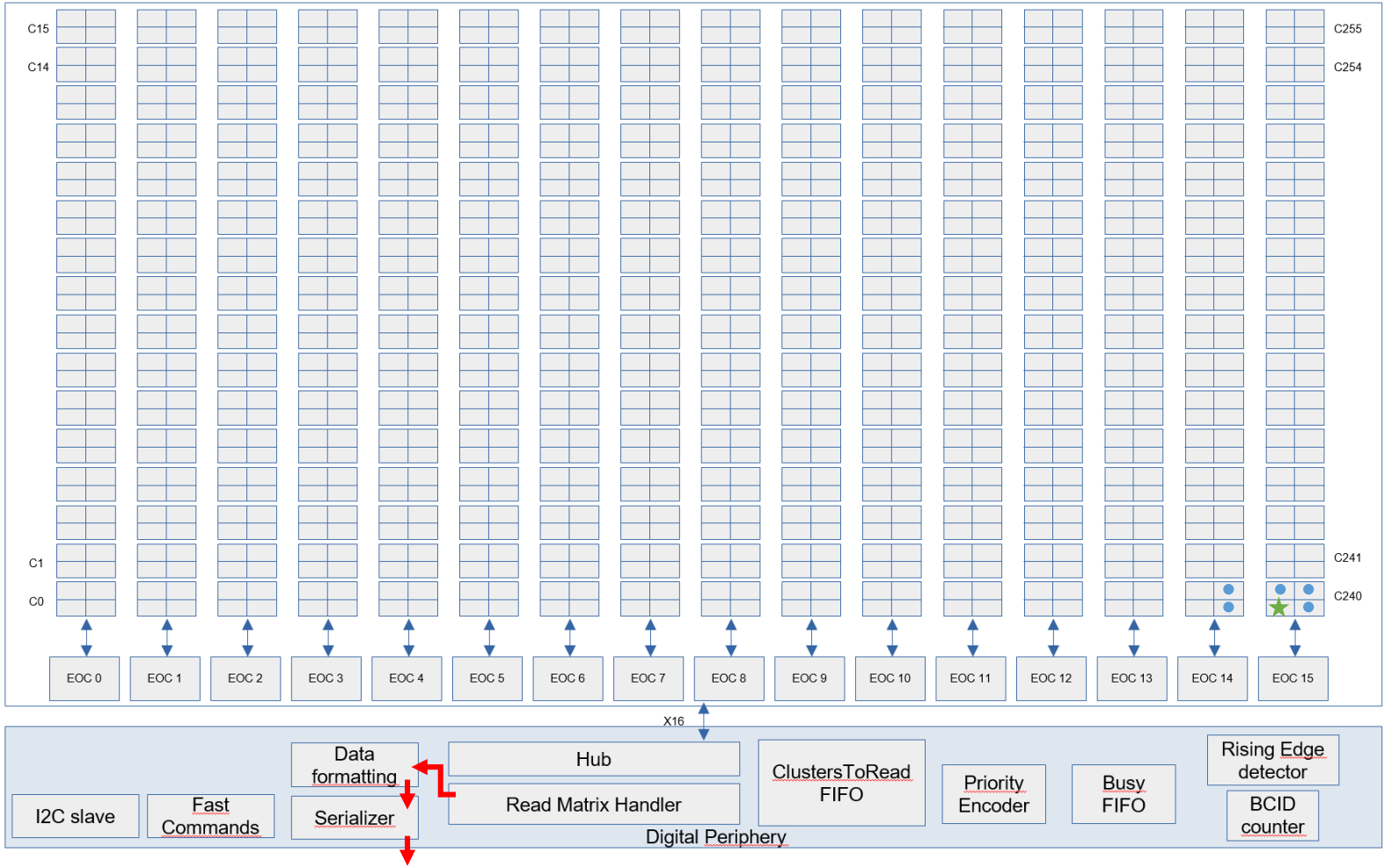
Architecture overview

1. Clusters send a busy signal to periphery when one of their pixels is hit
2. Periphery assign these signals to a timestamp (=BCID)
3. Addresses of clusters to read are retrieved from these signals and stored in a FIFO
4. Read requests are sent to relevant clusters
5. Cluster send pixels data (ADC+TDC) when it is adressed



Architecture overview

1. Clusters send a busy signal to periphery when one of their pixels is hit
2. Periphery assign these signals to a timestamp (=BCID)
3. Addresses of clusters to read are retrieved from these signals and stored in a FIFO
4. Read requests are sent to relevant clusters
5. Cluster send pixels data (ADC+TDC) when it is adressed
6. Data are formatted then serialized before being output of the chip (BCID, pixID, TDCdata, ADCdata)



Only matrix readout is setup for now, no data formatting, no fast commands, no I2C slave, no serializing…