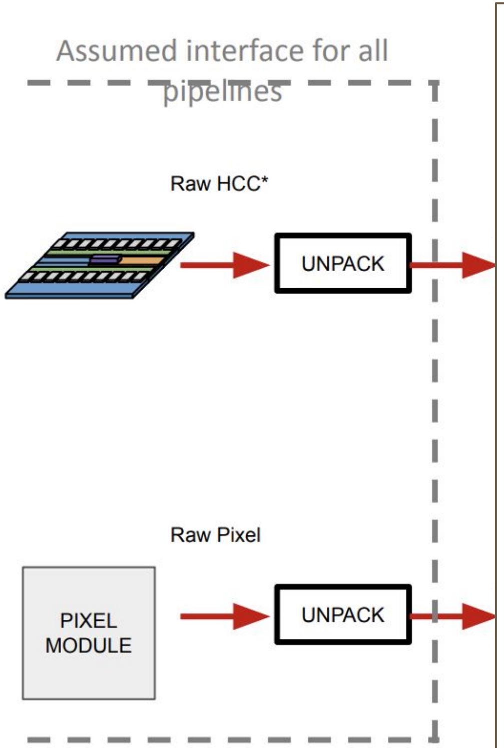
Pixel & Strip Data Format Proposal

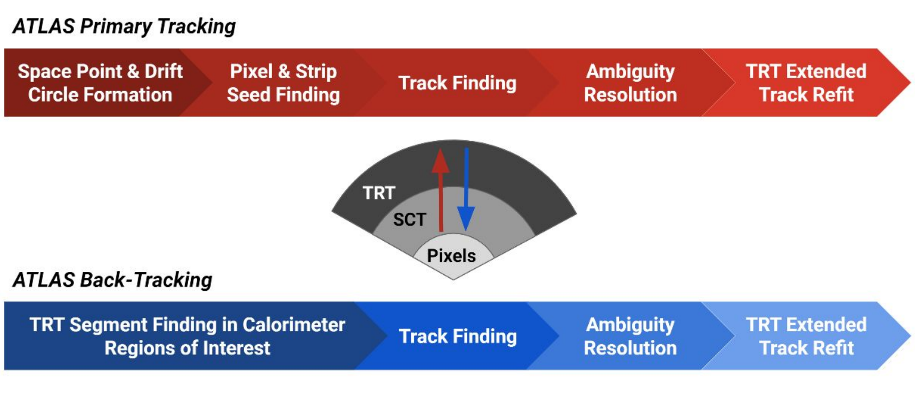
Haider
April 15th, 2024

Introduction

- For EF tracking, we need to establish the full data connection with the detector at some point
    - But this is **not critical** at the moment
- Hence need to **abstract out an interface** that contains all the information that will be given to us in some format in some order
    - But this requires an understanding of what happens inside the detector
    - Trying to document the information that lead to this data format proposal
    - Ideally this interface is exactly the same between online/offline
- The information presented here is just **a first proposal** within the EF tracking community and is being iterated upon

What we think we need to do (in pictures)



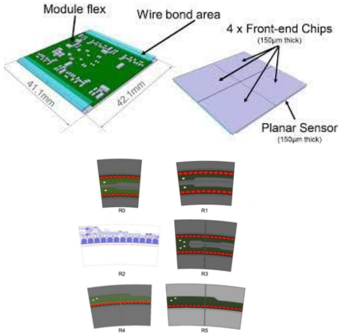


EF Tracking

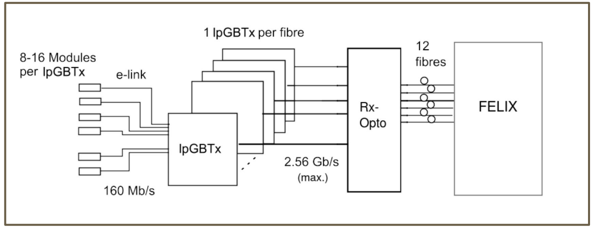
Different tracking chains

What we really (partly) need to do (in pictures)

Sensor



ITK readout pipeline



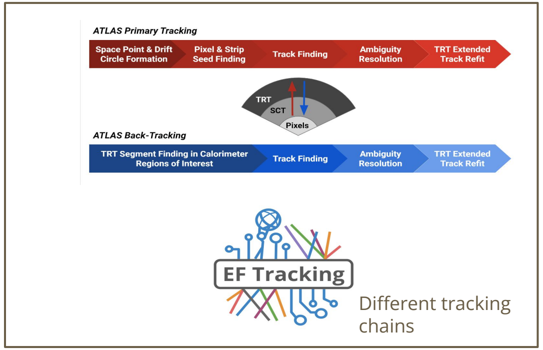
This is my artistic impression. Definitely not even fully representative, as it is missing DCS, calibrations, etc

Control Software*

ITK readout software

Interface glue

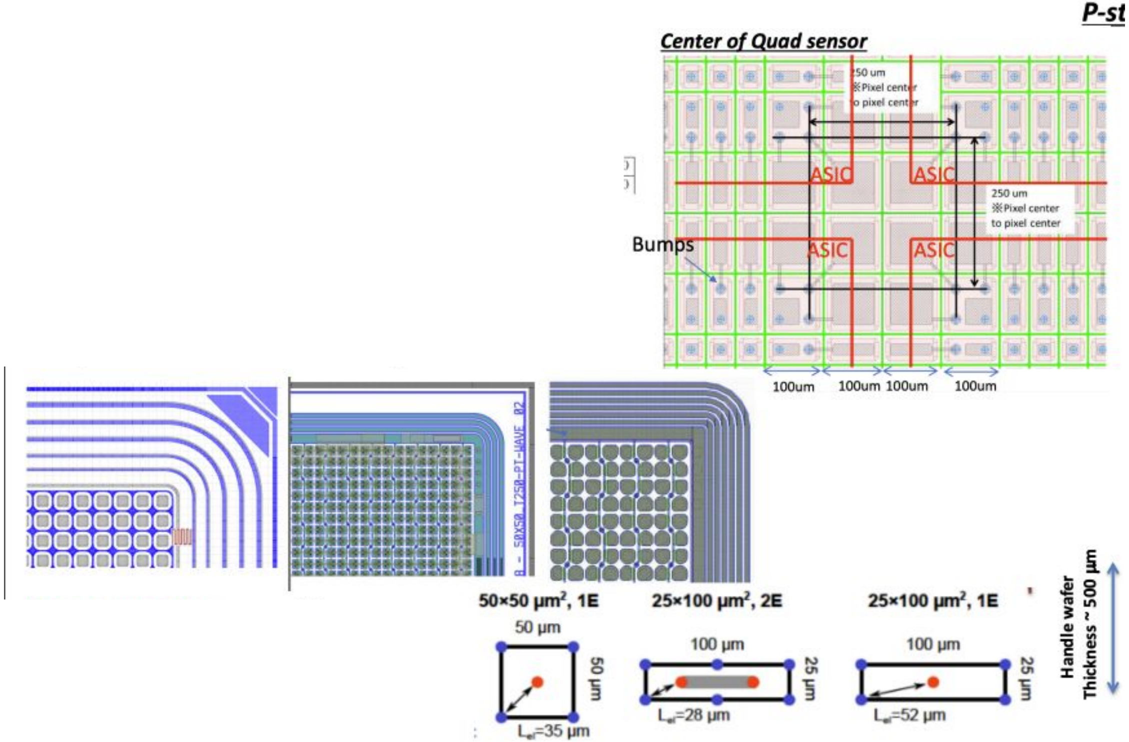
Host Code in Athena Framework



Different tracking chains

Pixel Sensor Geometry

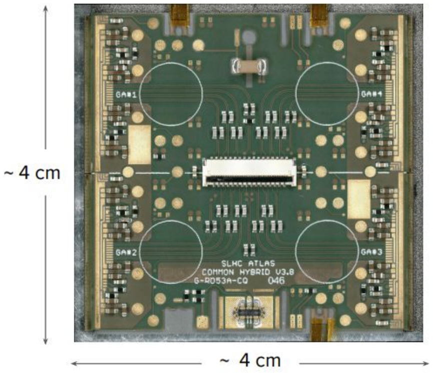
- Two types of pixel size, 50x50 and 25x100um


P-st
Center of Quad sensor
250 um ※Pixel center to pixel center
ASIC
ASIC
250 um ※Pixel center to pixel center
Bumps
ASIC
ASIC
100um
100um
100um
100um
50X50_T250-PT-WAVE_02
50×50 μm^2, 1E
25×100 μm^2, 2E
25×100 μm^2, 1E
50 µm
50 µm
L_{el}=35 µm
100 µm
25 µm
L_{el}=28 µm
100 µm
25 µm
L_{el}=52 µm
Handle wafer Thickness ~ 500 µm

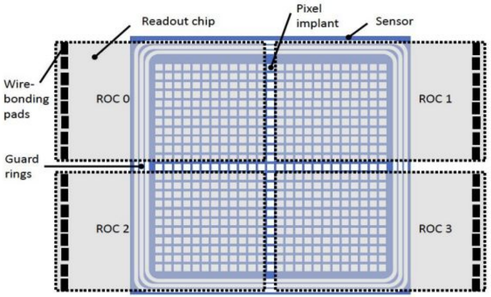
Pixel Module Geometry

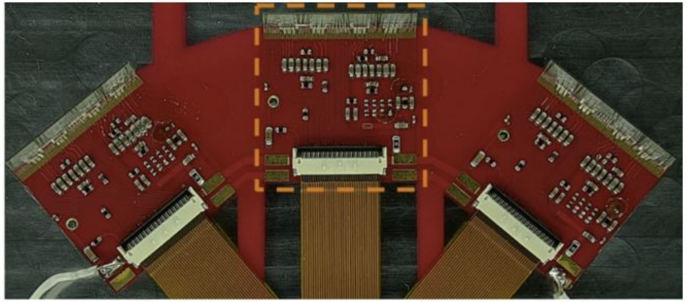
- Two types of modules

Example of the assembled RD53A module



Sketch of quad-module layout





Quad module

4 FE reading one silicon sensor

Triplet module

3 FE, each FE reading one separate silicon sensor

Few reasons why this matter

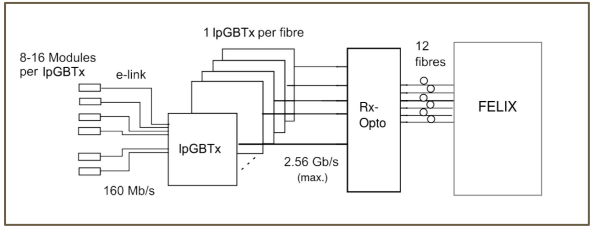
- Clustering needs to happen across a silicon sensor - particle will traverse the FE boundaries for a quad module
- Alignment should be performed for each silicon sensor and not FE

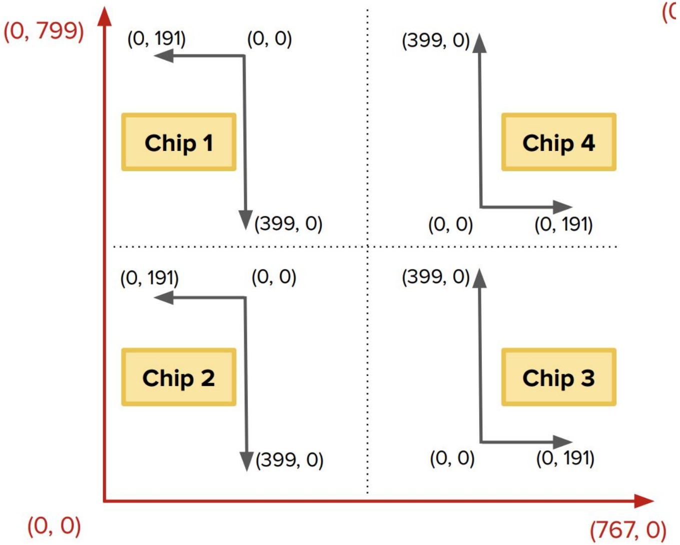
- **This boil down to the question “What defines a module/SiDetector Element in code? “**
    - My personal preference it should be one silicon sensor
    - Rest of the proposal is born from this preference

Few reasons why this is hard

- Mapping of FE across readout links are extremely complex
  - In certain layers, one link per FE
  - In other layers, one link for multiple FE
- FE orientation is not constant in a quad
  - Somewhere someone has to maintain this mapping when reading the links
- Either the readout software or some block will be needed to translate/group FE data into "module/SiDetectorElement"

ITK readout pipeline

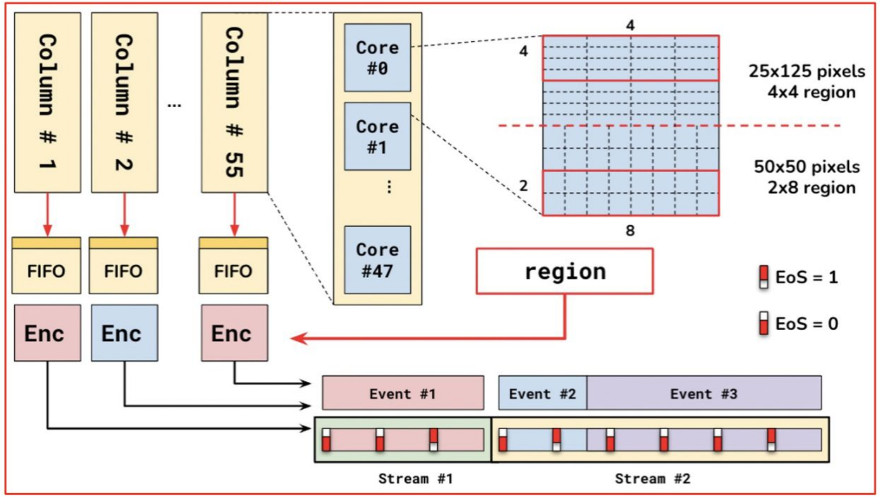




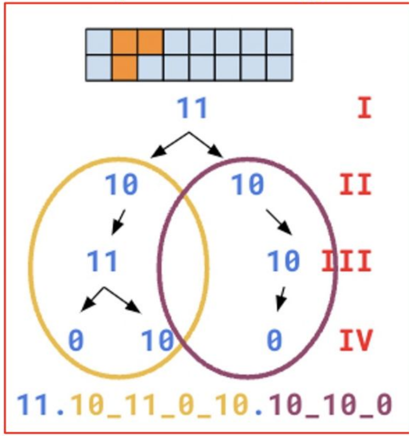
Pixel readout

- Pixel information is highly encoded
  - Binary compression from the FE and then a 2x8 cluster mapping in the second stage
  - It is not decided where this decoding will happen
- **What stage of decoding should EF tracking start at?**

Event building

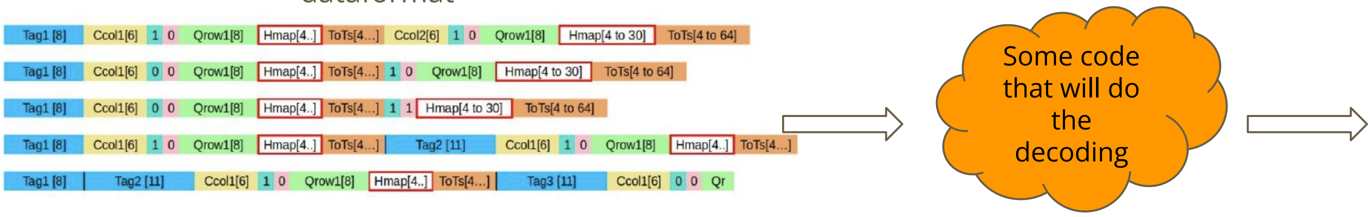


Binary encoding

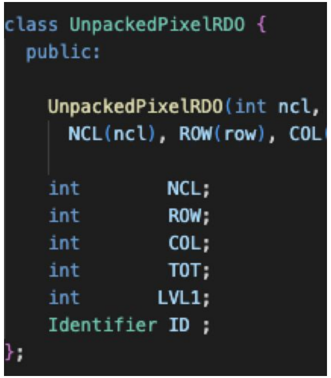


First proposal for pixel format

Highly encoded and compressed raw pixel dataformat


Tag1 [8] Ccol1[6] 1 0 Qrow1[8] Hmap[4..] ToTs[4...] Ccol2[6] 1 0 Qrow1[8] Hmap[4 to 30] ToTs[4 to 64]
Tag1 [8] Ccol1[6] 0 0 Qrow1[8] Hmap[4..] ToTs[4...] 1 0 Qrow1[8] Hmap[4 to 30] ToTs[4 to 64]
Tag1 [8] Ccol1[6] 0 0 Qrow1[8] Hmap[4..] ToTs[4...] 1 1 Hmap[4 to 30] ToTs[4 to 64]
Tag1 [8] Ccol1[6] 1 0 Qrow1[8] Hmap[4..] ToTs[4...] Tag2 [11] Ccol1[6] 1 0 Qrow1[8] Hmap[4..] ToTs[4...]
Tag1 [8] Tag2 [11] Ccol1[6] 1 0 Qrow1[8] Hmap[4..] ToTs[4...] Tag3 [11] Ccol1[6] 0 0 Qr
Some code that will do the decoding

“Physics data” for pixel

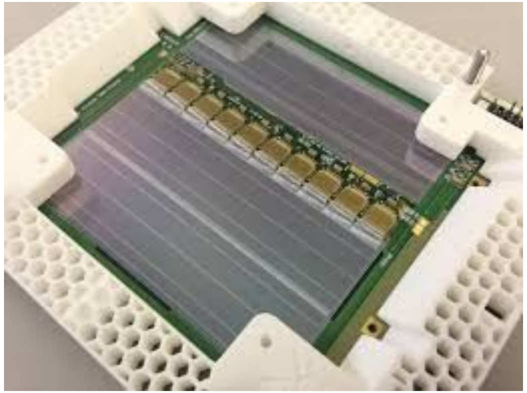


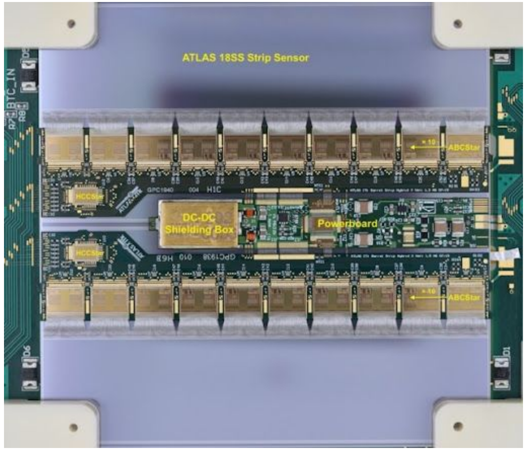
```
class UnpackedPixelRDO {
  public:

    UnpackedPixelRDO(int ncl,
      NCL(ncl), ROW(row), COL

    int          NCL;
    int          ROW;
    int          COL;
    int          TOT;
    int         LVL1;
    Identifier ID ;
};
```

- EF Pixel cluster should start with the UnpackedPixelRDO format - **Matches current offline format**
    - Row/Col is defined across a silicon sensor
- **If ITK software doesn’t unpack to this, extra converters will be needed in the middle - TBD where they will be hosted**
    - This allows for parallel development on EF chain and on the interface as more things are defined across different sub-systems

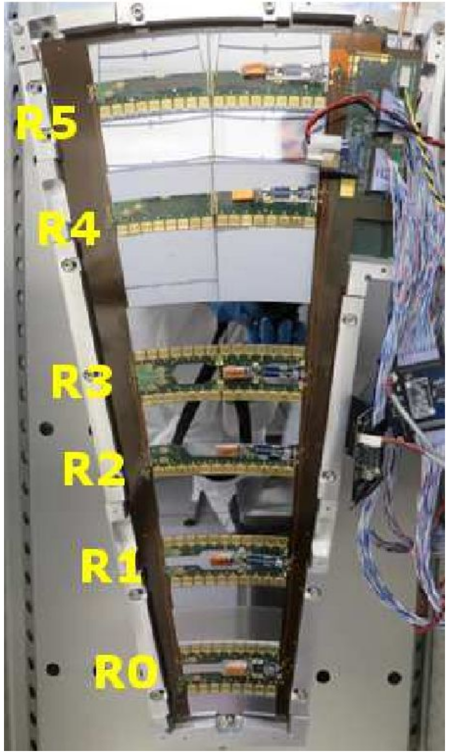
Strip Module Geometry

- Many types of modules





Barrel

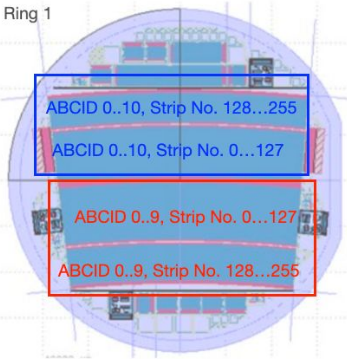


Endcap

Strip readout

- It is simpler and more complex at the same time
  - 10-21 FE Chips (ABCStar) for each silicon sensor
  - Communication happens through a HCC chip which gathers information from a row of FE chip on a “hybrid”
  - Each ABCStar data is split into 2 ‘streams’ - Upper half vs lower half of sensor are read by the same FE with different strip IDs

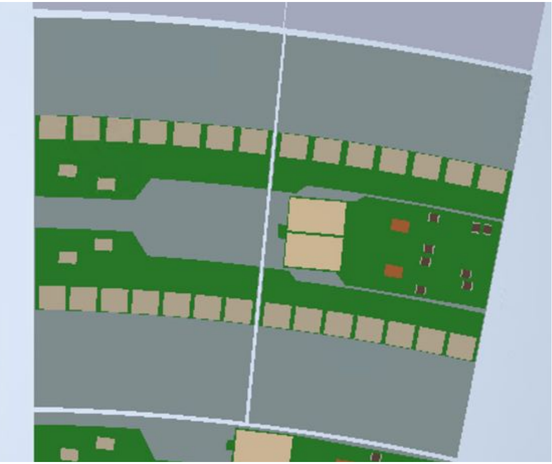
R1 is the simplest EC structure with 1 sensor & 2 HCC. Already has 4 “substructure”


Ring 1
ABCID 0..10, Strip No. 128…255
ABCID 0..10, Strip No. 0…127
ABCID 0..9, Strip No. 0…127
ABCID 0..9, Strip No. 128…255

HCC 0, 11 ABCs

HCC 1 , 10 ABCs

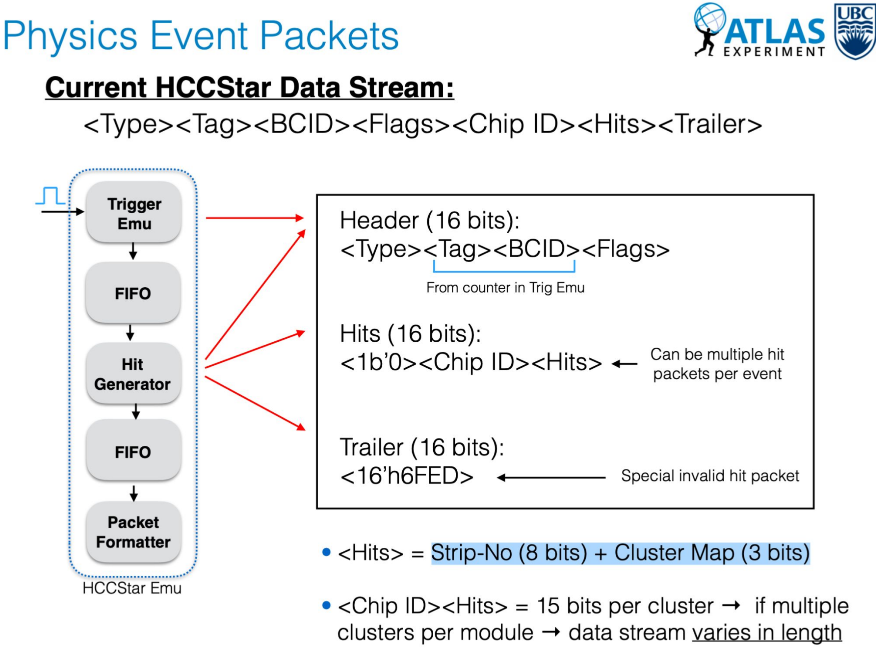
On the other hand, R3 is 2 Si sensors, with 4 HCC



First proposal for strip format

- Strip data format is ‘simple’
  - A lot of pre-processing is done already
  - HCC has a pre-clustering block that does cluster across an ABCStar FE
  - We should not undo this step

- Hence, first proposal is to keep the format ~ similar as FE output for EF Tracking chain



Physics Event Packets

Current HCCStar Data Stream:

<Type><Tag><BCID><Flags><Chip ID><Hits><Trailer>

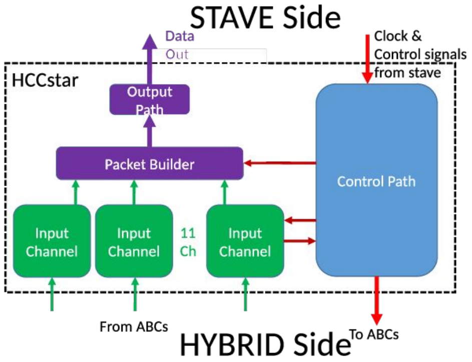
Header (16 bits):
<Type><Tag><BCID><Flags>
From counter in Trig Emu

Hits (16 bits):
<1b’0><Chip ID><Hits> ← Can be multiple hit packets per event

Trailer (16 bits):
<16’h6FED> ← Special invalid hit packet

- <Hits> = Strip-No (8 bits) + Cluster Map (3 bits)
- <Chip ID><Hits> = 15 bits per cluster → if multiple clusters per module → data stream varies in length

Preclustering block inside HCC



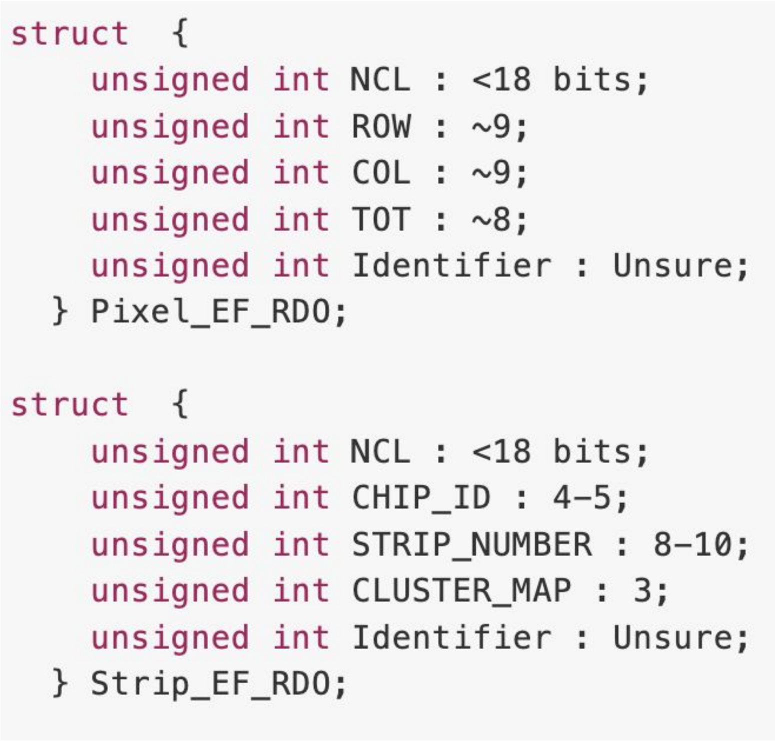
FE output from HCC

Few reasons why this is hard

- Current offline has a different Strip RDO EDM
  - Curent, Hit Format = <Strip #> + hit map
  - New proposed, Hit Format = <Strip #> + cluster size
  - Can offline support/use this format?
- If we ask for information from “one sensor”, the strip chip ID field doesn’t have enough range to support all the FE IDs
  - Again either readout software reshuffles and adds extra info or an extra converters to organize
- ABCStar Stream 0/1 seem to be split by strip ID - corresponds to upper/lower half
  - 0-127 strip ID -> Stream 0 & 128-255 strip ID -> Steam 1
  - Complicates computation of local coordinates
  - **TO BE CONFIRMED**

Conclusions

- A **first proposal** for data format and organization for EF tracking chain
    - Allows for EF chain to continue development without need for precise definition of the interface to the detector
- Big question: Is this data format/organization possible in the readout software?
    - If not, a potential solution is to add intermediate steps in-front to organize data in this format
    - To be determined with all involved stakeholders



```
struct  {
    unsigned int NCL : <18 bits;
    unsigned int ROW : ~9;
    unsigned int COL : ~9;
    unsigned int TOT : ~8;
    unsigned int Identifier : Unsure;
  } Pixel_EF_RDO;

struct  {
    unsigned int NCL : <18 bits;
    unsigned int CHIP_ID : 4-5;
    unsigned int STRIP_NUMBER : 8-10;
    unsigned int CLUSTER_MAP : 3;
    unsigned int Identifier : Unsure;
  } Strip_EF_RDO;
```