A new LCIO *(more)* CALICE oriented object ?

A. Irles, IFIC – CSIC / Universidad de Valencia

30th June 2021

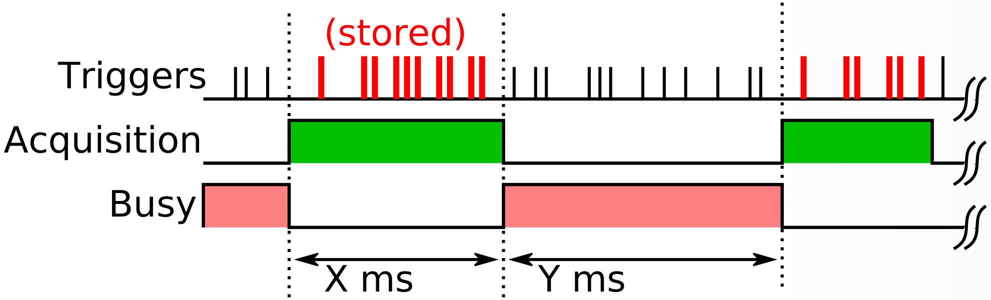
CALICE ASIC data

- CALICE ASICs have a similar output / behaviour
- OMEGA chips (SKIROC, SPIROC, HARDROC…)
  - Analogue or (semi)digital
- What about other chips ? (KLAUS?)
  - Input needed… here

Everything discussed here is applicable only to real raw-detector data.

CALICE ASIC data

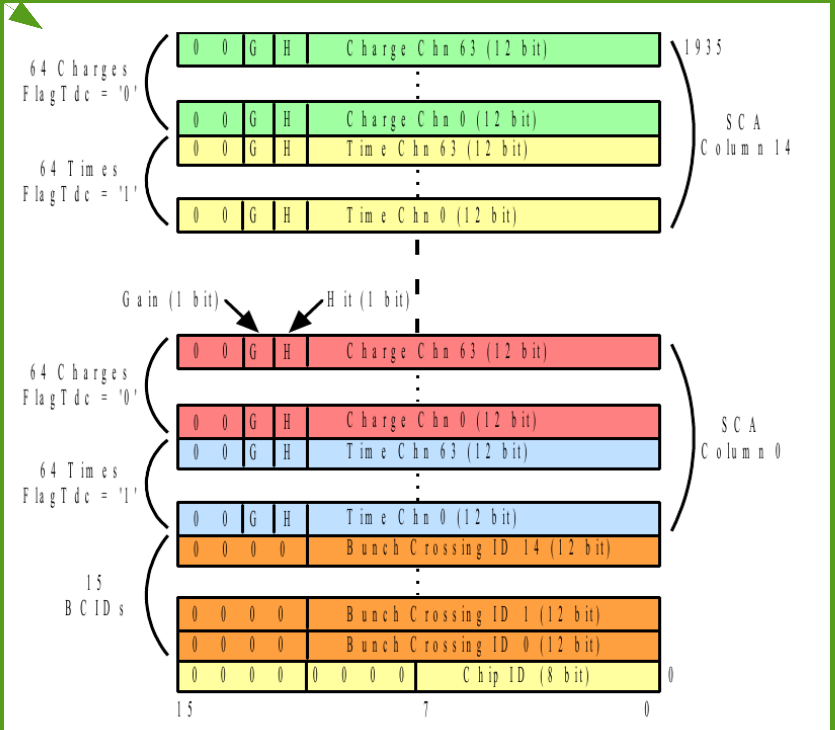
- **Spill based readout**
  - Open acquisition windows for O(1)ms
  - Store several events (15/16…) in each acquisition (*or readout cycle or spill… different naming but same/similar concepts)*
  - Every CHIP / MODULE / READOUT UNIT (data aggregator, lda, core module…) is independent of the others
  - Self-triggered cells
- **The data come unsorted**
  - **The full chips send the data before than the others.**
  - **Zero event building is done by the CHIPs.**
- **We may (or not) have external information on triggers (time/position if TLU and/or telescope)**



Raw data output format

- The **analogue** OMEGA chips providel:
- for every channel and memory cell
  - **Amplitude 1** (charge)
  - **Amplitude 2** (charge of fine time resolution TDC)
  - **1 bit for gain info**
  - **1 bit for trigger info** (self trigger)
- For every memory cell
  - **BCID** (time info… with granularity of 200ns – or similar)
- Global variables
  - Chip ID
  - Acquisition (or spill) number

SKIROC-Data 1



64 Charges FlagTdc = '0'
64 Times FlagTdc = '1'
15 BCIDs
0 0 G H Charge Chn 63 (12 bit)
0 0 G H Charge Chn 0 (12 bit)
0 0 G H Time Chn 63 (12 bit)
0 0 G H Time Chn 0 (12 bit)
1935
SCA Column 14
Gain (1 bit)
Hit (1 bit)
SCA Column 0
0 0 0 0 Bunch Crossing ID 14 (12 bit)
0 0 0 0 Bunch Crossing ID 1 (12 bit)
0 0 0 0 Bunch Crossing ID 0 (12 bit)
0 0 0 0 0 0 0 0 Chip ID (8 bit)
0
15
7
0

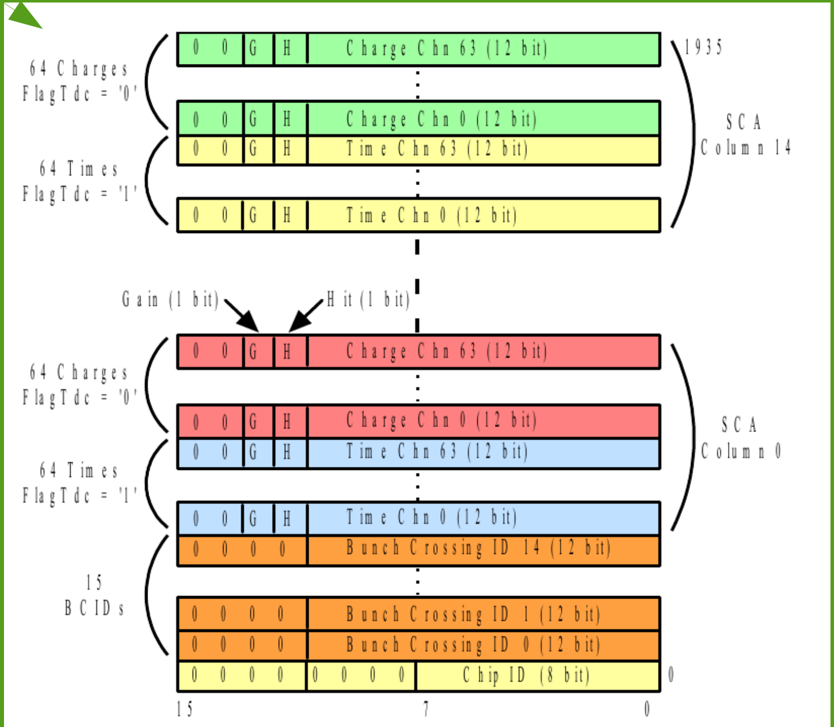
Figure 15: SKIROC2 chip memory mapping

Raw data output format

- The **analogue** OMEGA chips provide:
- for every channel and memory cell
  - **Amplitude 1** (charge)
  - **Amplitude 2** (charge of fine time resolution TDC)
  - **1 bit for gain info**
  - **1 bit for trigger info** (self trigger)
- For every memory cell
  - **BCID** (time info… with granularity of 200ns – or similar)
- Global variables
  - Chip ID
  - Acquisition (or spill) number

SKIROC-Data

1



64 Charges FlagTdc = '0'

0 0 G H Charge Chn 63 (12 bit) 1935

0 0 G H Charge Chn 0 (12 bit)

64 Times FlagTdc = '1'

0 0 G H Time Chn 63 (12 bit)

0 0 G H Time Chn 0 (12 bit)

SCA Column 14

Gain (1 bit)

Hit (1 bit)

64 Charges FlagTdc = '0'

0 0 G H Charge Chn 63 (12 bit)

0 0 G H Charge Chn 0 (12 bit)

64 Times FlagTdc = '1'

0 0 G H Time Chn 63 (12 bit)

0 0 G H Time Chn 0 (12 bit)

SCA Column 0

15 BCIDs

0 0 0 0 Bunch Crossing ID 14 (12 bit)

0 0 0 0 Bunch Crossing ID 1 (12 bit)

0 0 0 0 Bunch Crossing ID 0 (12 bit)

0 0 0 0 0 0 0 0 Chip ID (8 bit) 0

15 7 0

Figure 15: SKIROC2 chip memory mapping

Up to my knowledge, the HARDROC (semi-digital ASICs) data structure is very similar (but “lighter”)

event building (step1)

- The first step of the event building requires quick access to:
  - Cycle Number (acquisition number)
  - The BCID
- If an external reference is used (i.e. TLU) we will also use the external time information for event building

event building (step1)

- ▶ The first step of the event building requires quick access to:
  - Cycle Number (acquisition number)
  - The BCID
- ▶ If an external reference is used (i.e. TLU) we will also use the external time information for event building

(step2 & more)

- ▶ Access to the charge measurements (conversion from ADC/TDC to energy/time)
- ▶ And geometrical disposition of hits
- ▶ Tracking / Clustering / etc...

Current CALICE-oriented LCIO object

RawCalorimeterHit

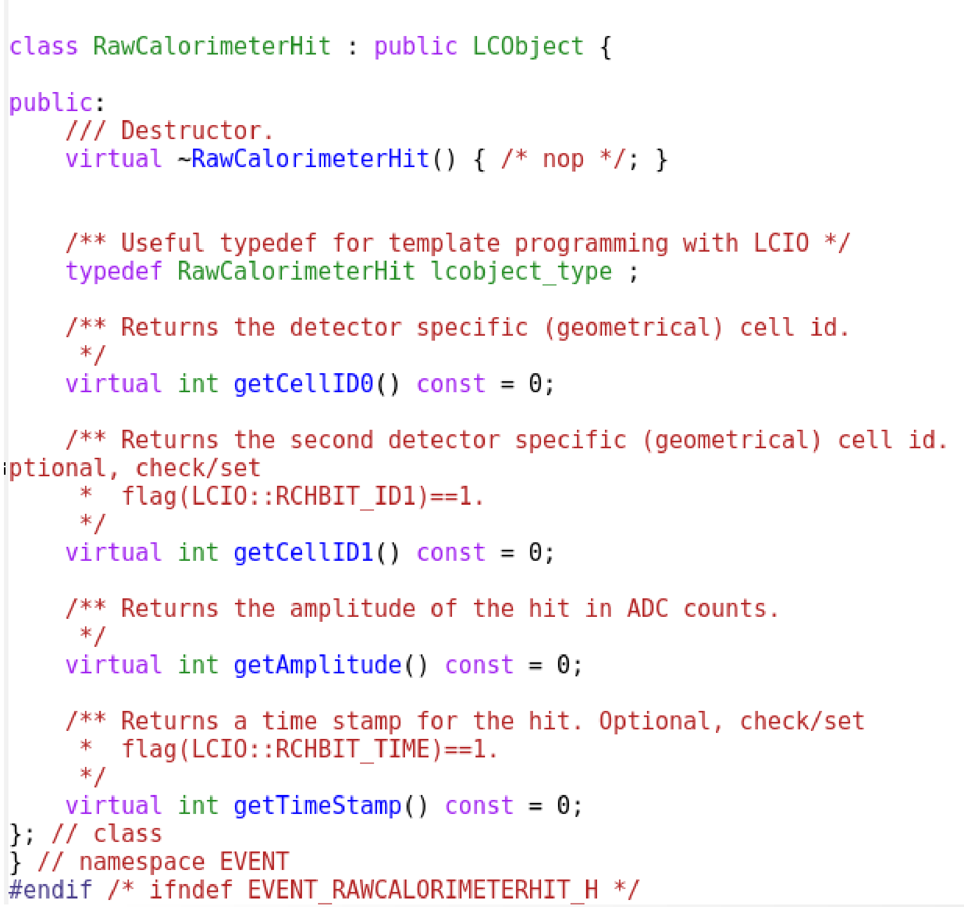
- ▶ Amplitude
  - Used for the charge
- ▶ TimeStamp (optional variable)
  - Used for the BCID
- ▶ GetCellID0 (and 1)
  - Used for the x/y/z or IJK or module/chip/chn/sca

- ▶ What about the second charge measurement?

  Or the TDC ?

  Or the gain / trigger bits ?

- ▶ **Only accessible if encoded in one of the previous variables**



```
class RawCalorimeterHit : public LCObject {

public:
    /// Destructor.
    virtual ~RawCalorimeterHit() { /* nop */; }


    /** Useful typedef for template programming with LCIO */
    typedef RawCalorimeterHit lcobject_type ;

    /** Returns the detector specific (geometrical) cell id.
     */
    virtual int getCellID0() const = 0;

    /** Returns the second detector specific (geometrical) cell id.
ptional, check/set
     *  flag(LCIO::RCHBIT_ID1)==1.
     */
    virtual int getCellID1() const = 0;

    /** Returns the amplitude of the hit in ADC counts.
     */
    virtual int getAmplitude() const = 0;

    /** Returns a time stamp for the hit. Optional, check/set
     *  flag(LCIO::RCHBIT_TIME)==1.
     */
    virtual int getTimeStamp() const = 0;
}; // class
} // namespace EVENT
#endif /* ifndef EVENT_RAWCALORIMETERHIT_H */
```

Current CALICE-oriented LCIO object

RawCalorimeterHit

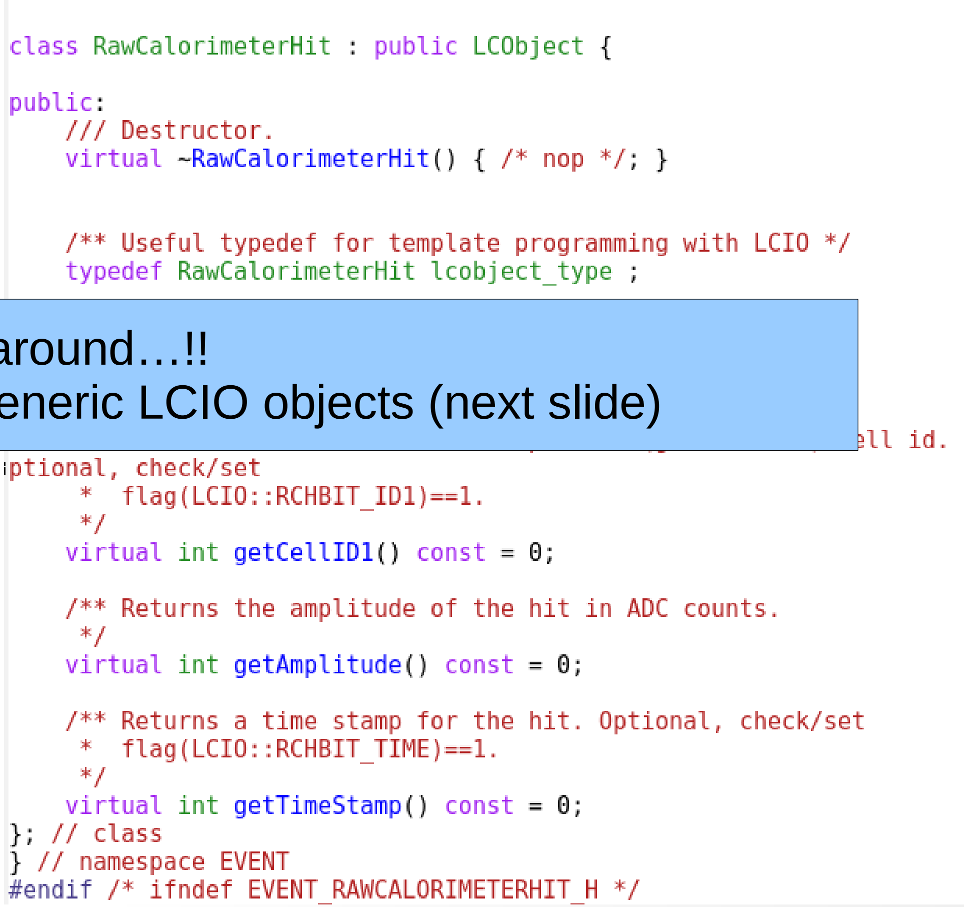
- Amplitude
  - Used for the charge
- TimeStamp (optional variable)
  - Used for the BCID
- GetCellID0 (
  - Used for th

Work-around…!!
Another solution is to use Generic LCIO objects (next slide)

- What about the second charge measurement?

  Or the TDC ?

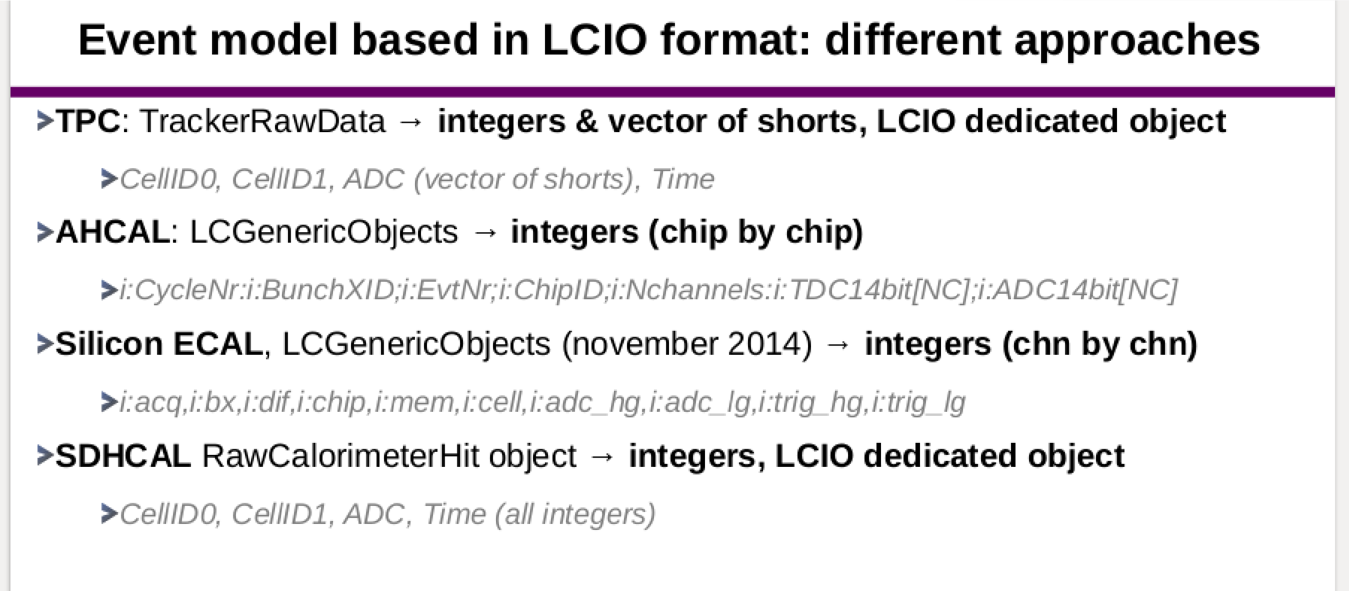
  Or the gain / trigger bits ?
- **Only accessible if encoded in one of the previous variables**



```
class RawCalorimeterHit : public LCObject {

public:
    /// Destructor.
    virtual ~RawCalorimeterHit() { /* nop */; }


    /** Useful typedef for template programming with LCIO */
    typedef RawCalorimeterHit lcobject_type ;
```

```
ell id.
ptional, check/set
     *  flag(LCIO::RCHBIT_ID1)==1.
     */
    virtual int getCellID1() const = 0;

    /** Returns the amplitude of the hit in ADC counts.
     */
    virtual int getAmplitude() const = 0;

    /** Returns a time stamp for the hit. Optional, check/set
     *  flag(LCIO::RCHBIT_TIME)==1.
     */
    virtual int getTimeStamp() const = 0;
}; // class
} // namespace EVENT
#endif /* ifndef EVENT_RAWCALORIMETERHIT_H */
```

LCIO objects used so far

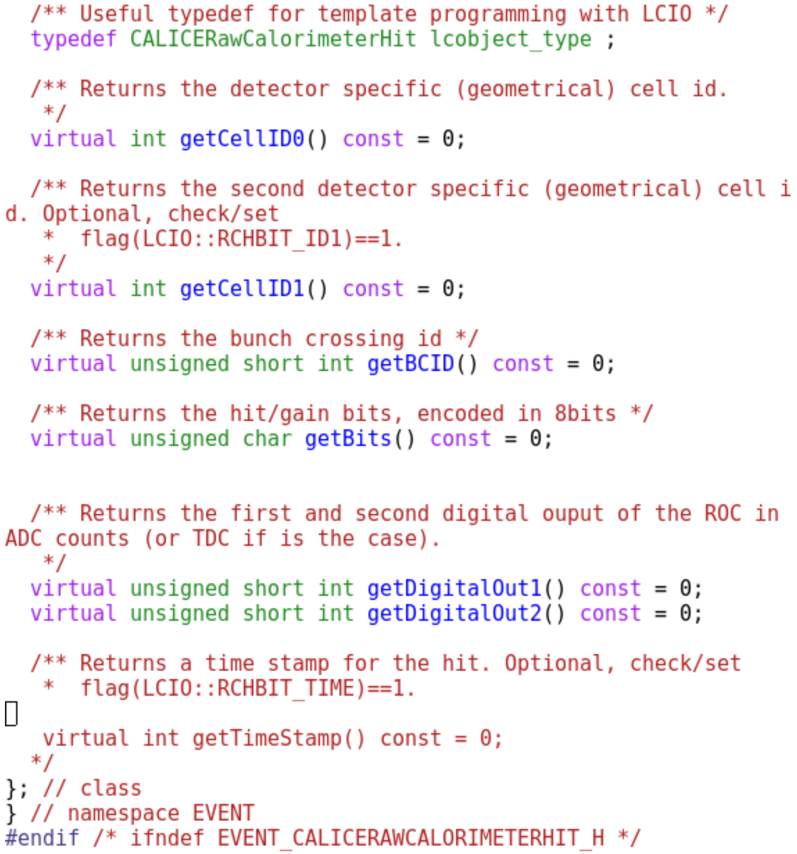


Event model based in LCIO format: different approaches

- **TPC**: TrackerRawData → **integers & vector of shorts, LCIO dedicated object**
  - *CellID0, CellID1, ADC (vector of shorts), Time*
- **AHCAL**: LCGenericObjects → **integers (chip by chip)**
  - *i:CycleNr:i:BunchXID;i:EvtNr;i:ChipID;i:Nchannels:i:TDC14bit[NC];i:ADC14bit[NC]*
- **Silicon ECAL**, LCGenericObjects (november 2014) → **integers (chn by chn)**
  - *i:acq,i:bx,i:dif,i:chip,i:mem,i:cell,i:adc_hg,i:adc_lg,i:trig_hg,i:trig_lg*
- **SDHCAL** RawCalorimeterHit object → **integers, LCIO dedicated object**
  - *CellID0, CellID1, ADC, Time (all integers)*

- Screenshot from a presentation from 2017 !!
- Dedicated objects (instead of LCGeneric objects) are:
  - ~Twice faster (disk writing!)
  - ~Twice more compressed (file size!)

Proposal 1



```
  /** Useful typedef for template programming with LCIO */
  typedef CALICERawCalorimeterHit lcobject_type ;

  /** Returns the detector specific (geometrical) cell id.
   */
  virtual int getCellID0() const = 0;

  /** Returns the second detector specific (geometrical) cell i
d. Optional, check/set
   *  flag(LCIO::RCHBIT_ID1)==1.
   */
  virtual int getCellID1() const = 0;

  /** Returns the bunch crossing id */
  virtual unsigned short int getBCID() const = 0;

  /** Returns the hit/gain bits, encoded in 8bits */
  virtual unsigned char getBits() const = 0;


  /** Returns the first and second digital ouput of the ROC in
ADC counts (or TDC if is the case).
   */
  virtual unsigned short int getDigitalOut1() const = 0;
  virtual unsigned short int getDigitalOut2() const = 0;

  /** Returns a time stamp for the hit. Optional, check/set
   *  flag(LCIO::RCHBIT_TIME)==1.
▯
   virtual int getTimeStamp() const = 0;
  */
}; // class
} // namespace EVENT
#endif /* ifndef EVENT_CALICERAWCALORIMETERHIT_H */
```

- ▶ Dedicated “lighter” objects (unsigned short int.. instead of int)
- ▶ New:
  - BCID
  - DigitalOutputs 1 and 2
  - Bits
- ▶ We keep:
  - CellID (for all geometric information)
- ▶ Possible variation: we keep the option of having an external TimeStamp (TLU, for example).

With this object,
for the event building we only use the
BCID information.
No further encoding needed.

Proposal 2

- Keep the same RawCalorimeterHit
- But add few variables more to it:
  - At least one more amplitude → that can be used for a second charge measurement or a TDC measurement
- The gain / trigger bits would have to be encoded inside other variables
  - For example in the cellID… possible but maybe not optimal.

Final comments

- The RawCalorimeterHit is a CALICE object
  - Only used by CALICE (or detector groups)
  - Not used in ILD simulations (they use SimCalorimeterHits + digitisiation etc… = CalorimeterHits)
  - Backward compatibility issues… would only affect us.
- There is some activity now happening in the LCIO package which will probably end up in a new (tag) version of LCIO
  - Willingess from the ILD soft expert to perform changes.

- We are dealing with raw data real events…
  - Spill based (not event based)
  - Unsorted, self-triggered.
- If we decide for a full redesign of the LCIO object (i.e. proposal 1 type) we are not forced to change the name…

- **The most important question: are the different group needs covered by the proposals in this talk ?**
  - Here I am assuming that digital is a “subset” of analogue (raw-data-wise)