G2p Helicity Setup

Chao Gu

Outline

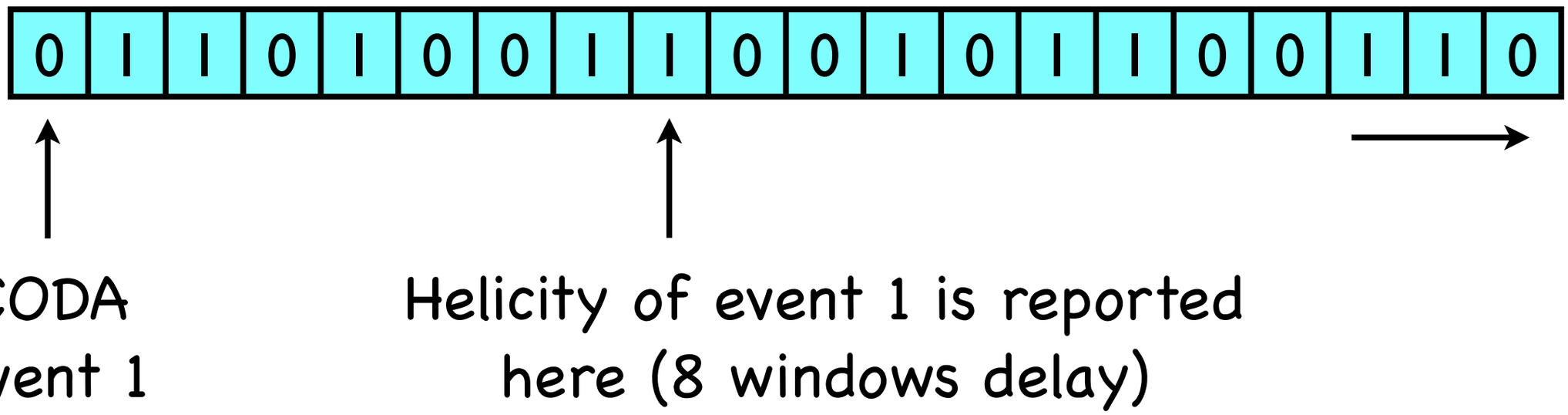
- Helicity Signal Scheme
- Hardware Setup
- Decode Package
- Charge Asymmetry Test
- Appendix

Helicity Signal Scheme

Back to Outline

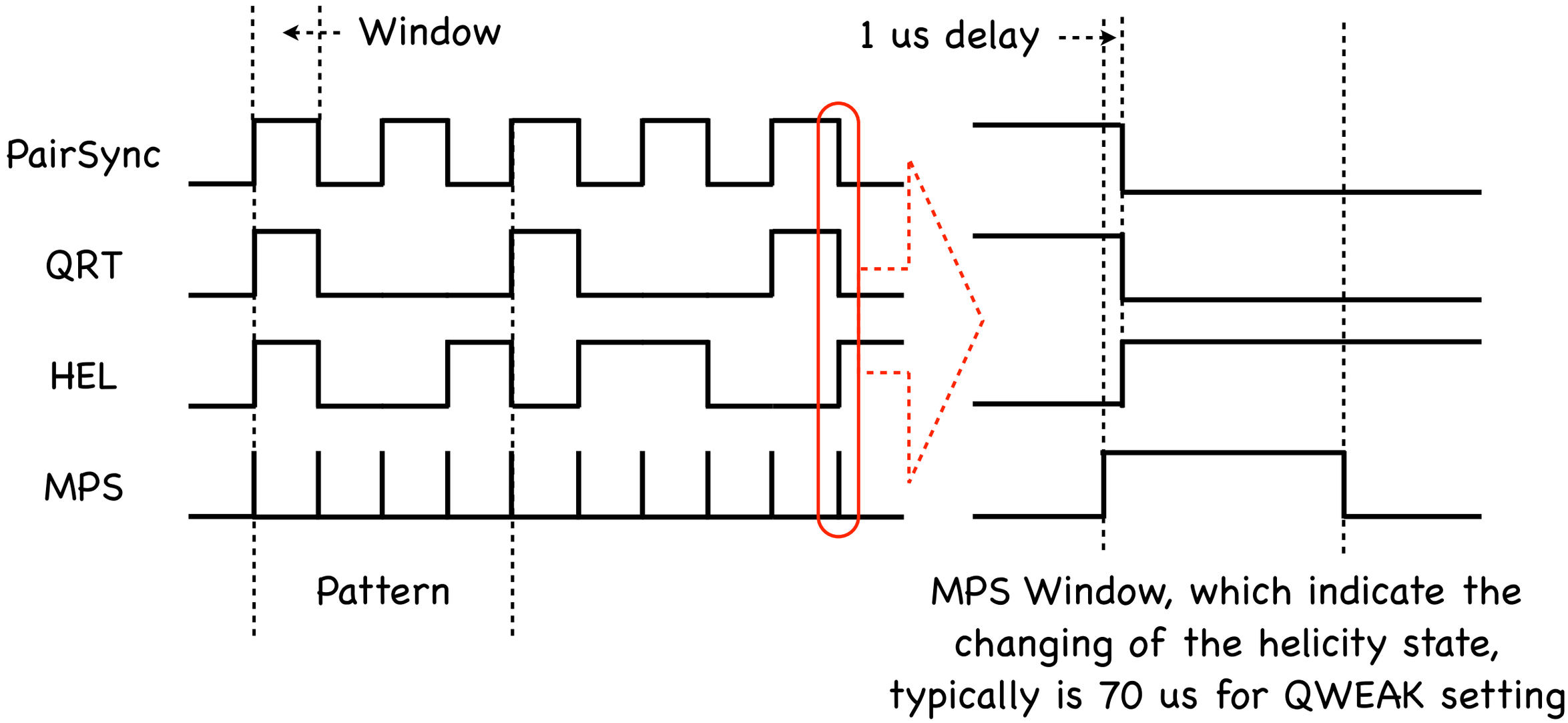
QWEAK Helicity Setting

- QWEAK runs with a helicity flip rate 960.02 Hz
- g2p registers CODA events at a rate up to 6 kHz
- The helicity reported from the helicity board is not the actual helicity of the beam at that time: delayed by 8 windows



Helicity Signal Scheme

- Four helicity signal: Helicity, QRT, MPS & PairSync
- The relationship of these 4 signals are shown in this plot:

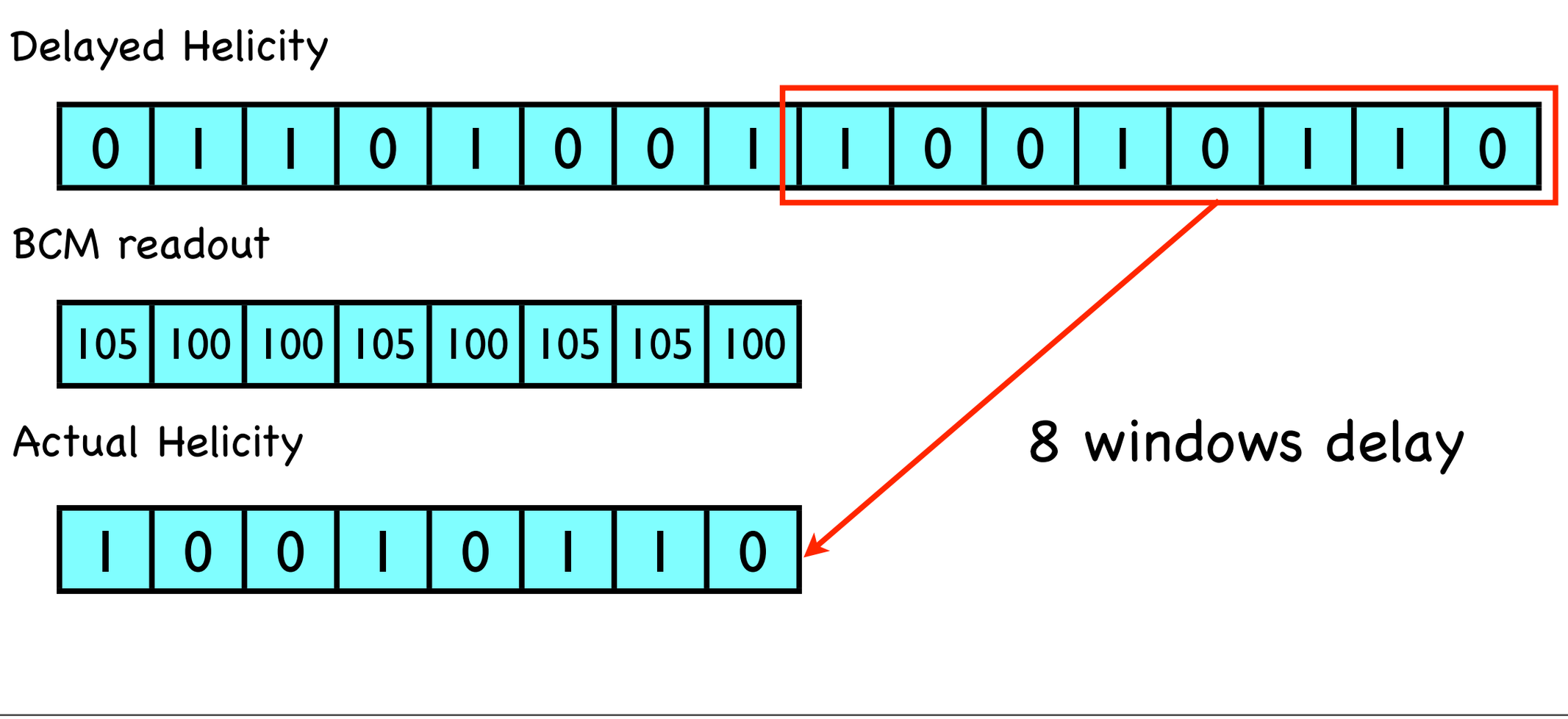


Helicity Signal Scheme: Definitions

- Helicity: indicate the polarization of the electron beam
- Window: a time region in which the helicity is stable, typically 971.65 us for QWEAK setting
- Pattern: to minimize system error, helicity is always changing by some symmetric sequence (like quartet +--+, octet +--+-++-, ...), which is called pattern, the pattern is composed by several helicity windows
- QRT: indicate the first window of a helicity pattern
- Polarity: the sign (positive or negative) of the first helicity window in each pattern
- MPS: indicate the changing time of the helicity state, in a MPS window, the helicity state is not well defined

Helicity Signal Scheme

- Readout data change with actual helicity (like BCM, Trigger, L1A, ..., will take BCM as example)
- Readout data is stored with delayed helicity
- Need to regenerate the actual helicity from the delayed helicity

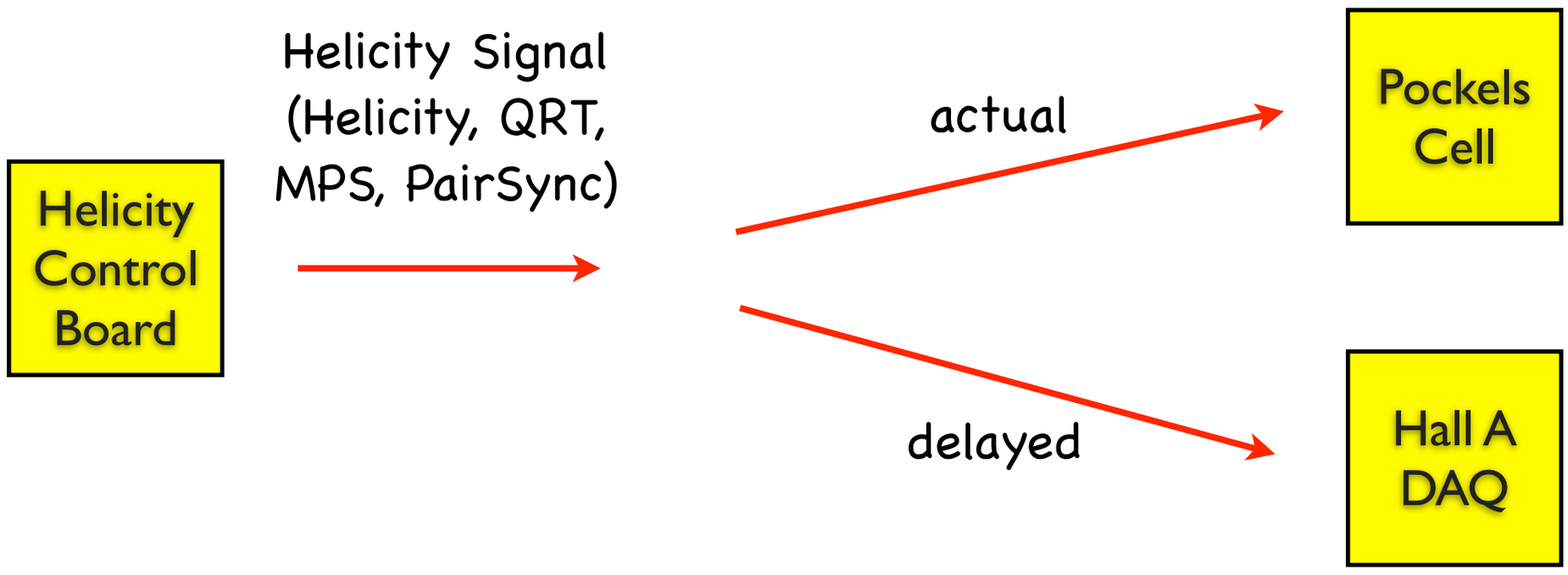


Hardware Setup

Back to Outline

Hardware Setup

- Helicity Signal is generated by Helicity Control Board installed at injector building
- Helicity board send actual helicity to injector to flip the real polarization of the electron beam
- Helicity board send delayed helicity to experiment halls

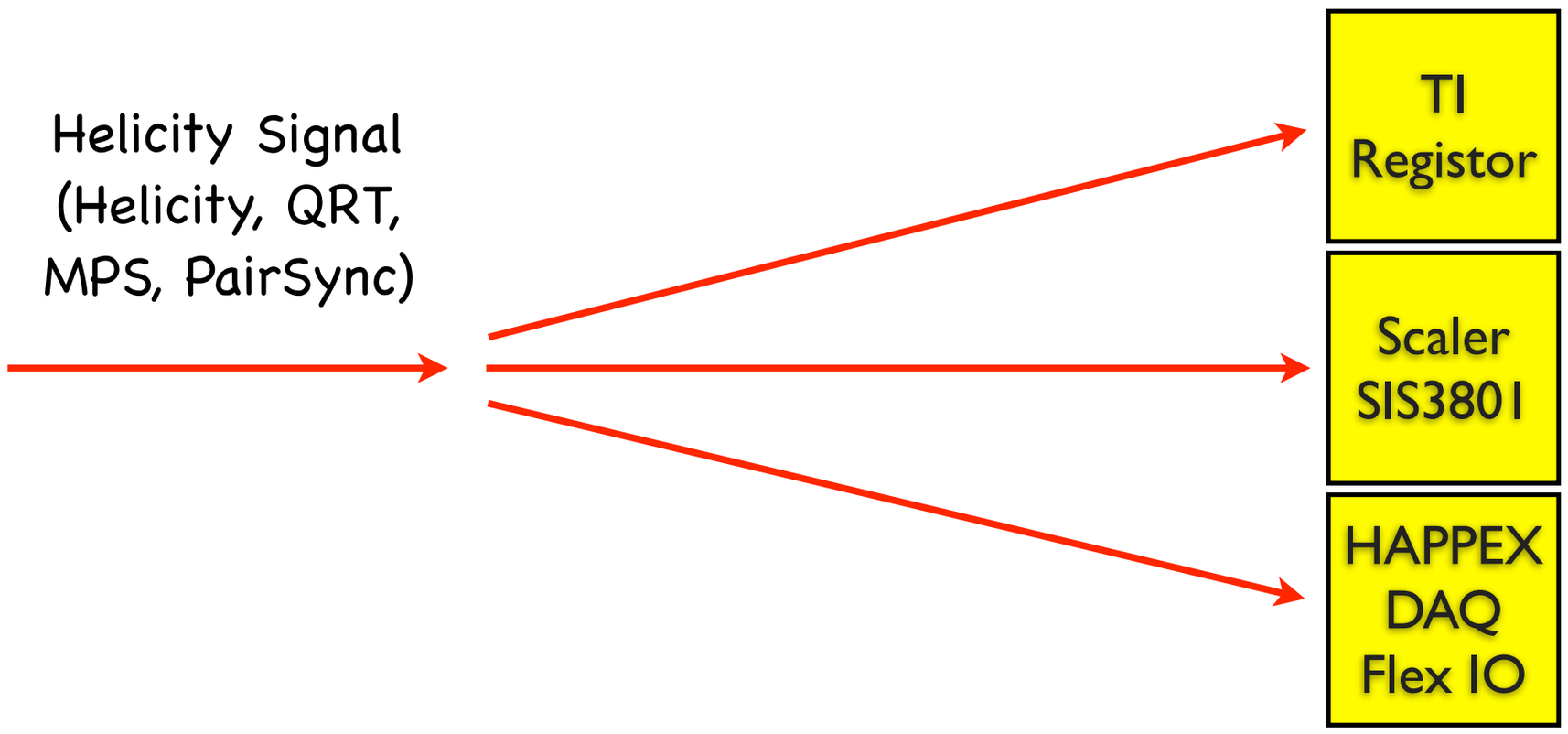


Hardware Setup

- Three ways to record helicity in g2p:
  - Trigger Interface Register(TIR): to record the helicity state for each physics event
  - Scaler Ring Buffer: to record the charge information sorted by the helicity state
    - HAPPEX DAQ: contain a ring buffer which is similar as scaler ring buffer
- Electronics setup for TIR and Scaler: see R. Michael's tech note for details
- HAPPEX Hardware Setup: I will skip this topic here, please see HAPPEX DAQ documents for details

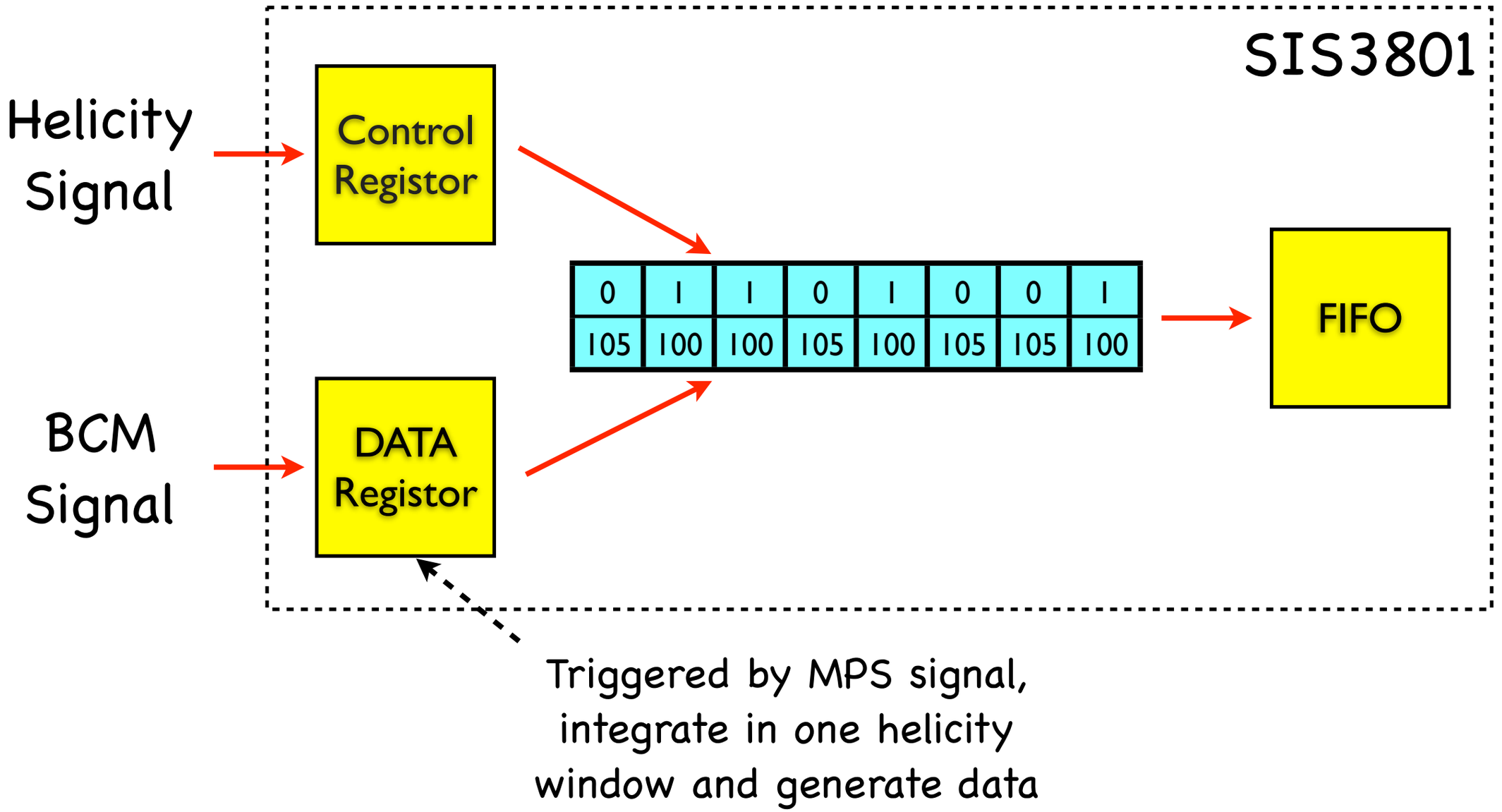
Hardware Setup: TIR

- 8 copies of helicity signal:
  - 3 copies to Left, right and 3rd arm's SIS3801 scaler
  - 3 copies to L, R & 3rd arm's TIR
  - 2 copies to L & R arm's HAPPEX DAQ
- TIR uses 4 registors to record the helicity signals
- The helicity information of physics event only recorded by TIR

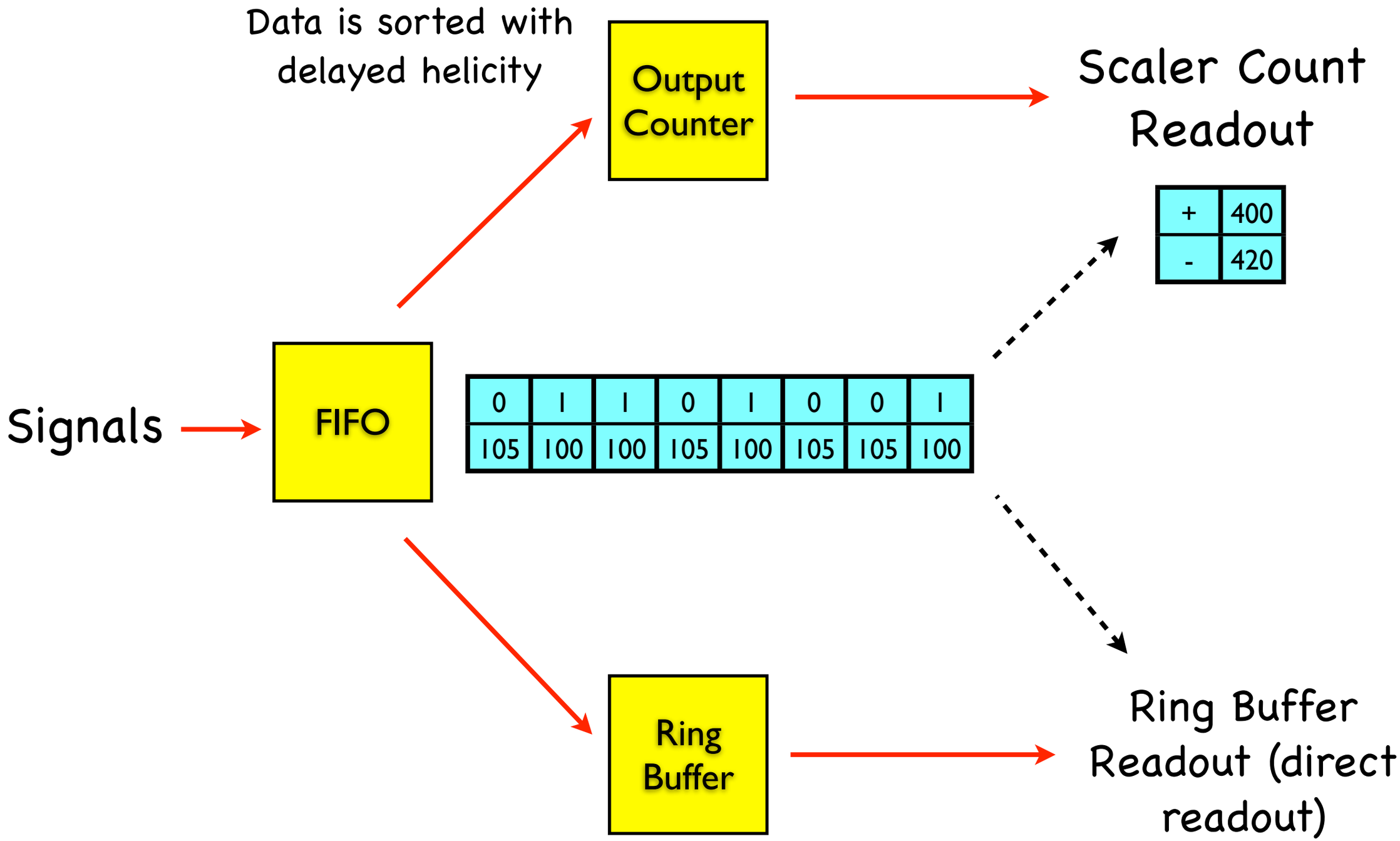


Hardware Setup: Scaler

- Electronics setup can be found here (by R. Michael) : http://hallaweb.jlab.org/equipment/daq/qweak_helicity.html (or in Appendix)

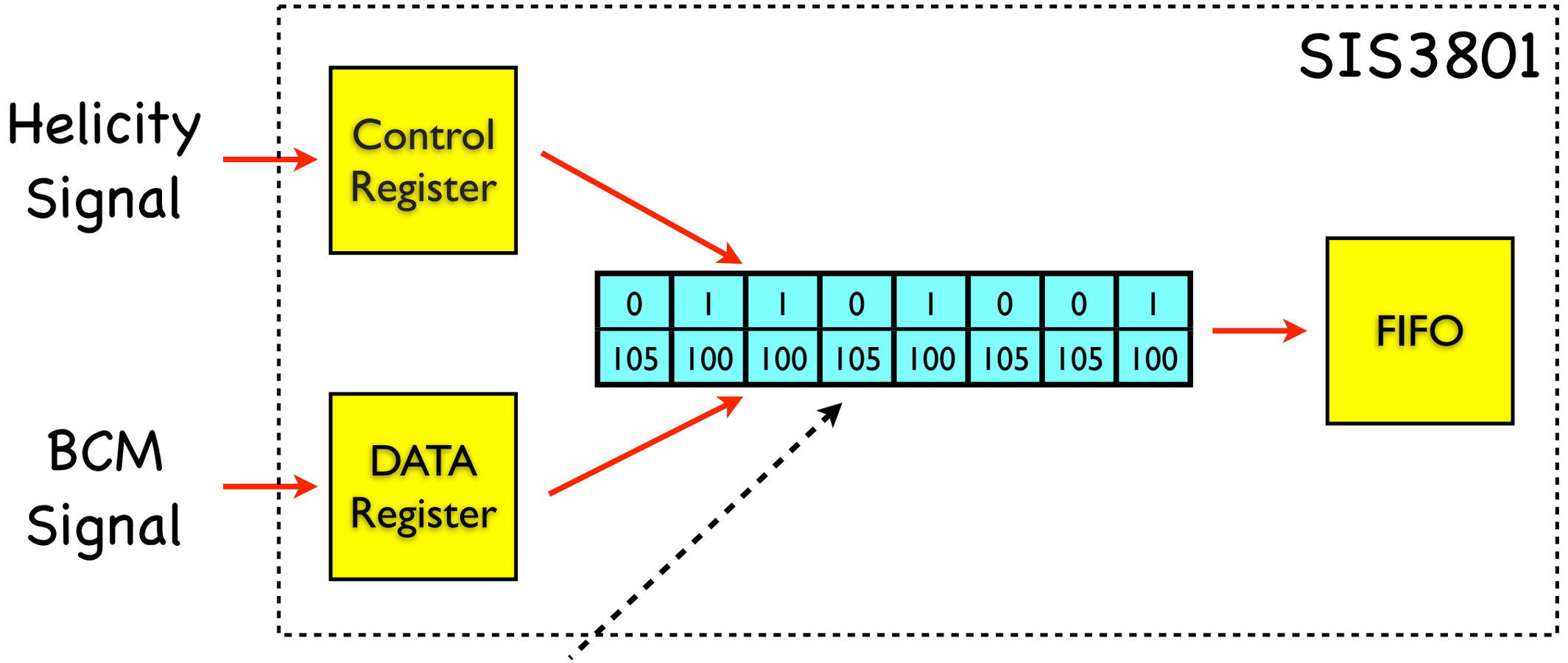


Hardware Setup: Scaler



Hardware Setup: Scaler

- One problem found during commission



The delayed helicity and BCM is not aligned inside SIS3801, it should be delayed by 8 windows but actually it is 7 windows

0	1	1	0	1	0	0	1
105	100	100	105	100	105	105	100

	0	1	1	0	1	0	0
105	100	100	105	100	105	105	100

Decode Package

Back to Outline

Decoder: Pseudo-random Helicity Generator

- The polarity of each helicity pattern is generated by a 30 bits shift registor, which is a pseudo-random sequence

Polarity sequence generated by the pseudo-random number generator

0
1
0
0
0
1
1
1

Each bit in this sequence will be expended with the helicity pattern

0	1	1	0
1	0	0	1
0	1	1	0
0	1	1	0
0	1	1	0
1	0	0	1
1	0	0	1
1	0	0	1

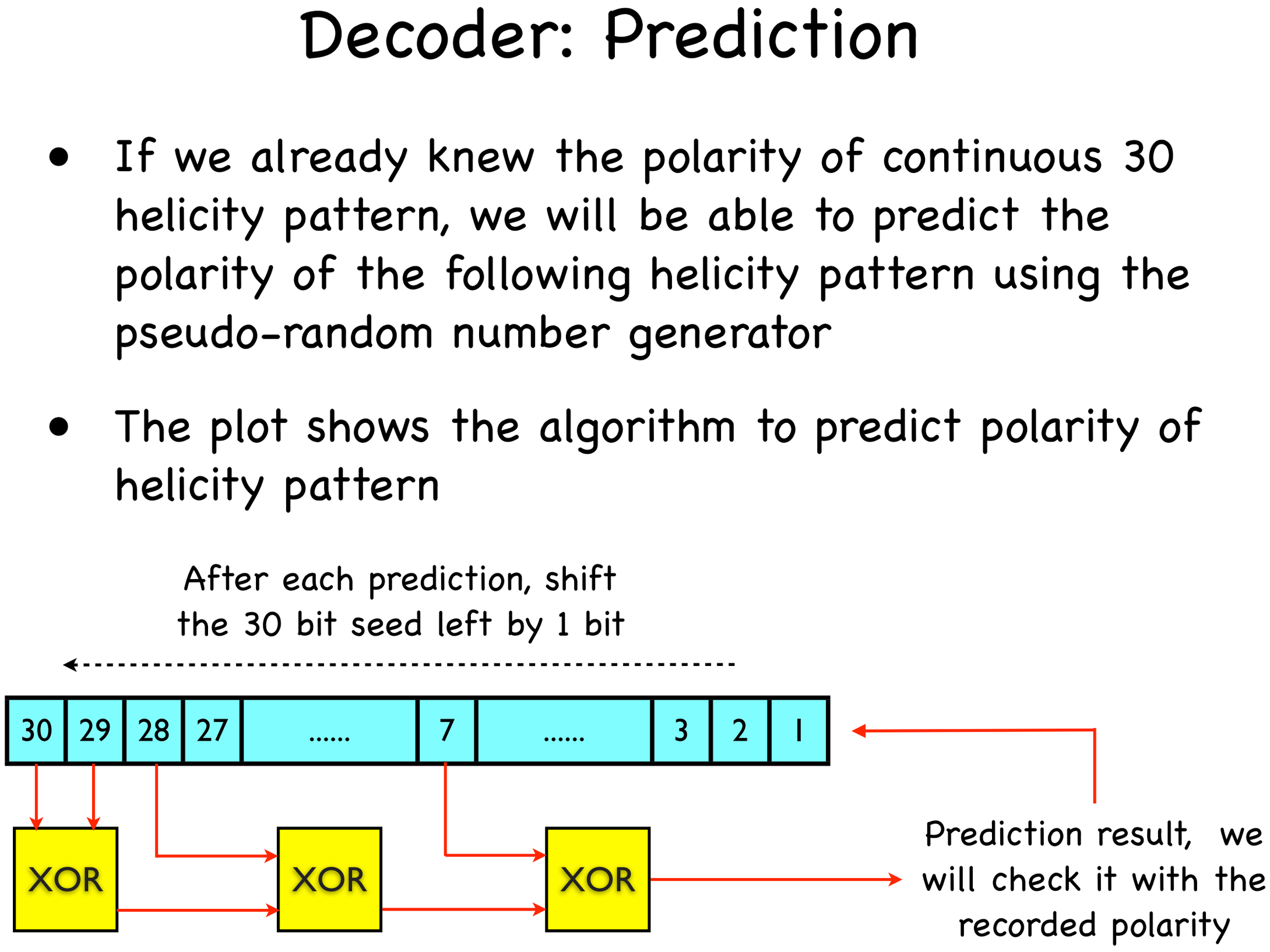
0	1	1	0
1	0	0	1
0	1	1	0
0	1	1	0
0	1	1	0
1	0	0	1
1	0	0	1
1	0	0	1

0	1	1	0	1	0	0	1	0	1	1	0	0	1	1	0	0	1	1	0	1	0	0	1

Connect all of the helicity patterns to generate the final helicity flipping sequence

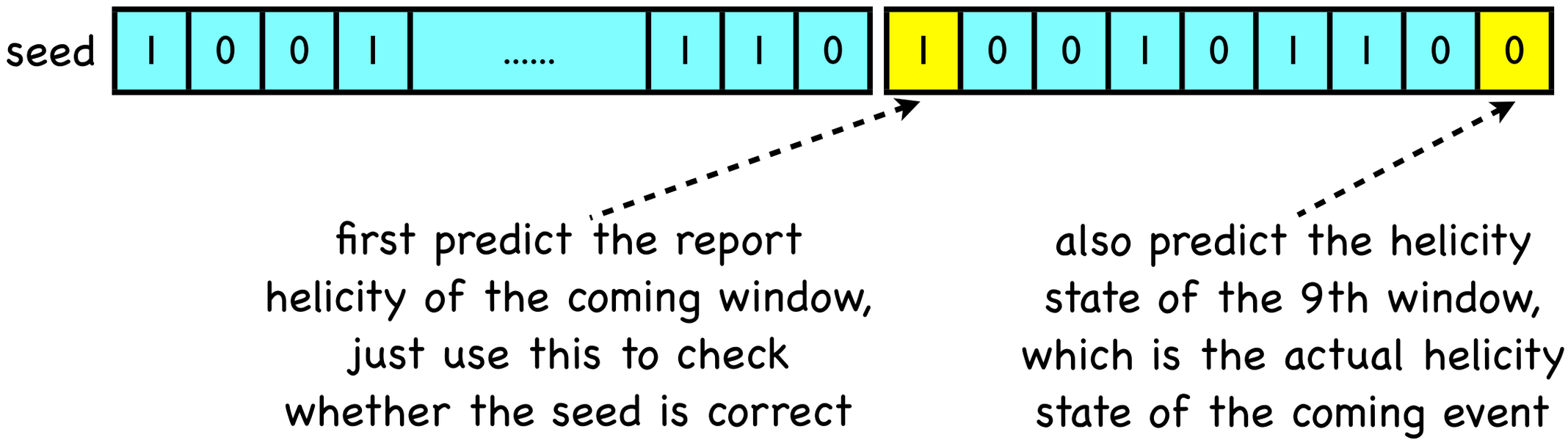
Decoder: Prediction

- If we already knew the polarity of continuous 30 helicity pattern, we will be able to predict the polarity of the following helicity pattern using the pseudo-random number generator
- The plot shows the algorithm to predict polarity of helicity pattern



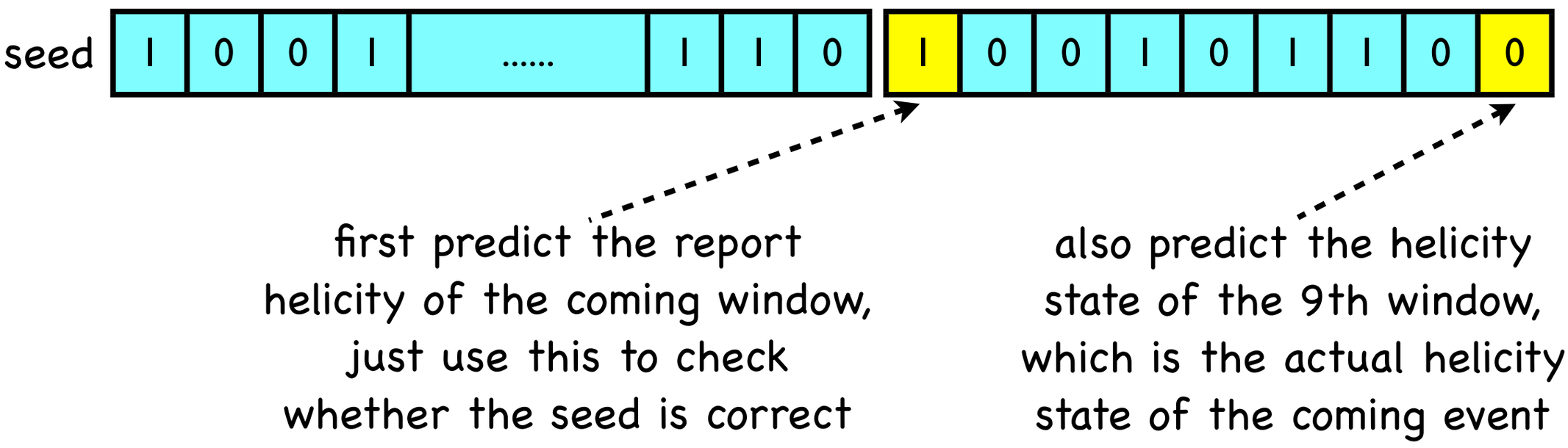
Decoder: Prediction

- Since the report helicity is delayed by eight windows, the basic idea of the decoder is to gather 30 helicity patterns and regenerate the pseudo-random seed, then use the seed to predict the actual helicity
- Will do 2 times of prediction for each helicity window after we got the random seed:



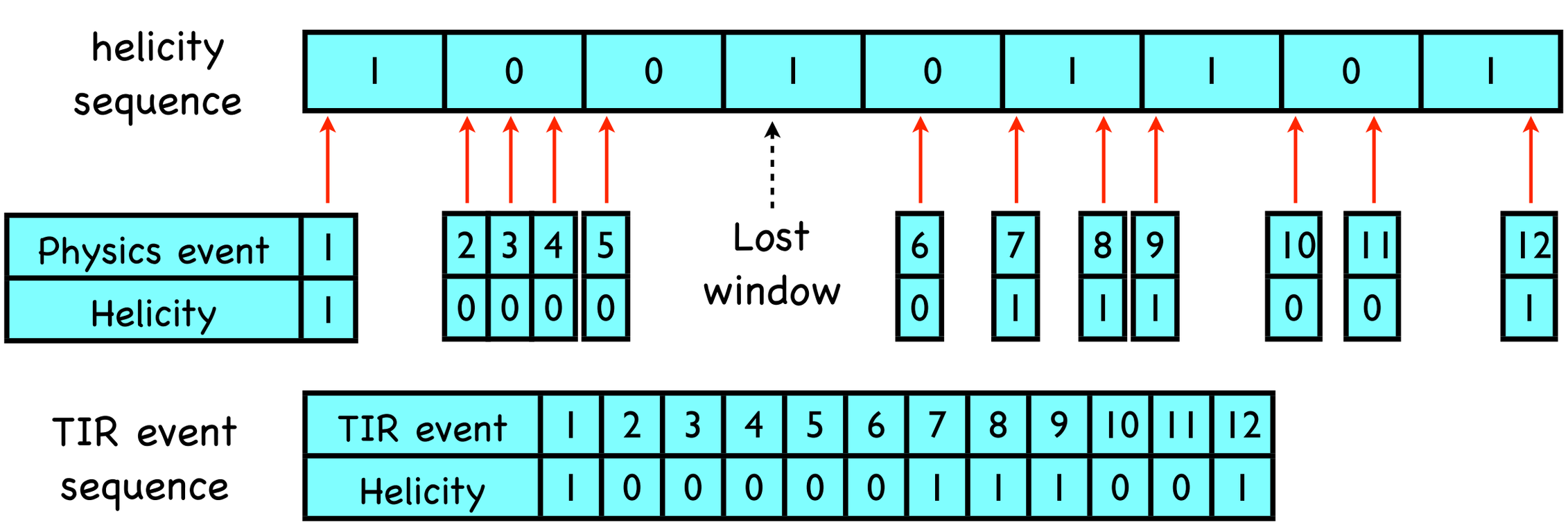
Decoder: Ring Buffer

- The scaler ring buffer is triggered by MPS signal, it will only record one event for each helicity window, and should not lose any helicity window if normal
- Helicity information in scaler ring buffer automatically satisfies the requirement of prediction
- No special treatment is required here, just process the prediction as described in previous slide



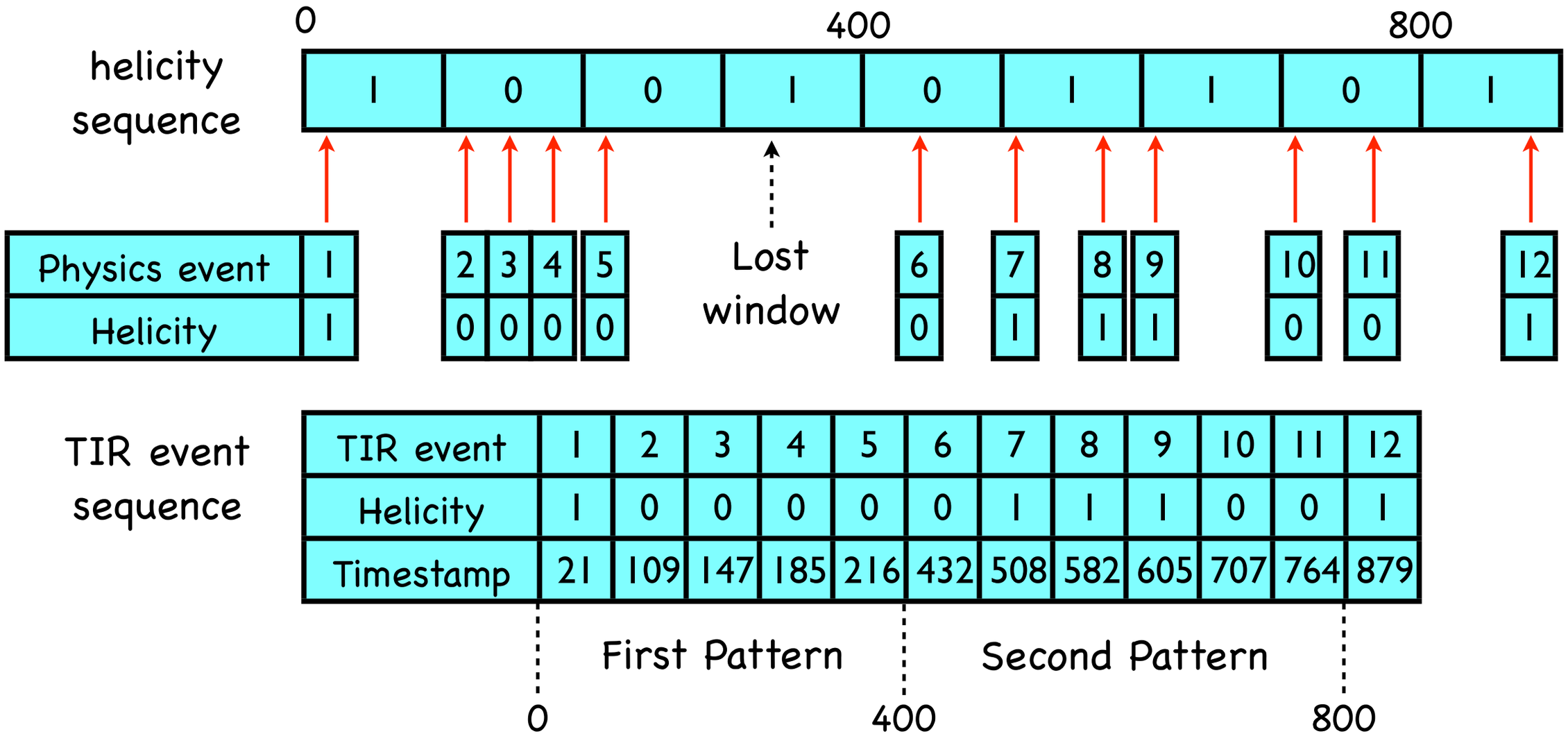
Decoder: TIR

- TIR is triggered by physics events, it may record several events in one helicity window, and may lose some helicity windows when there is no physics trigger
- Direct prediction as described in previous slides will FAIL



Decoder: TIR

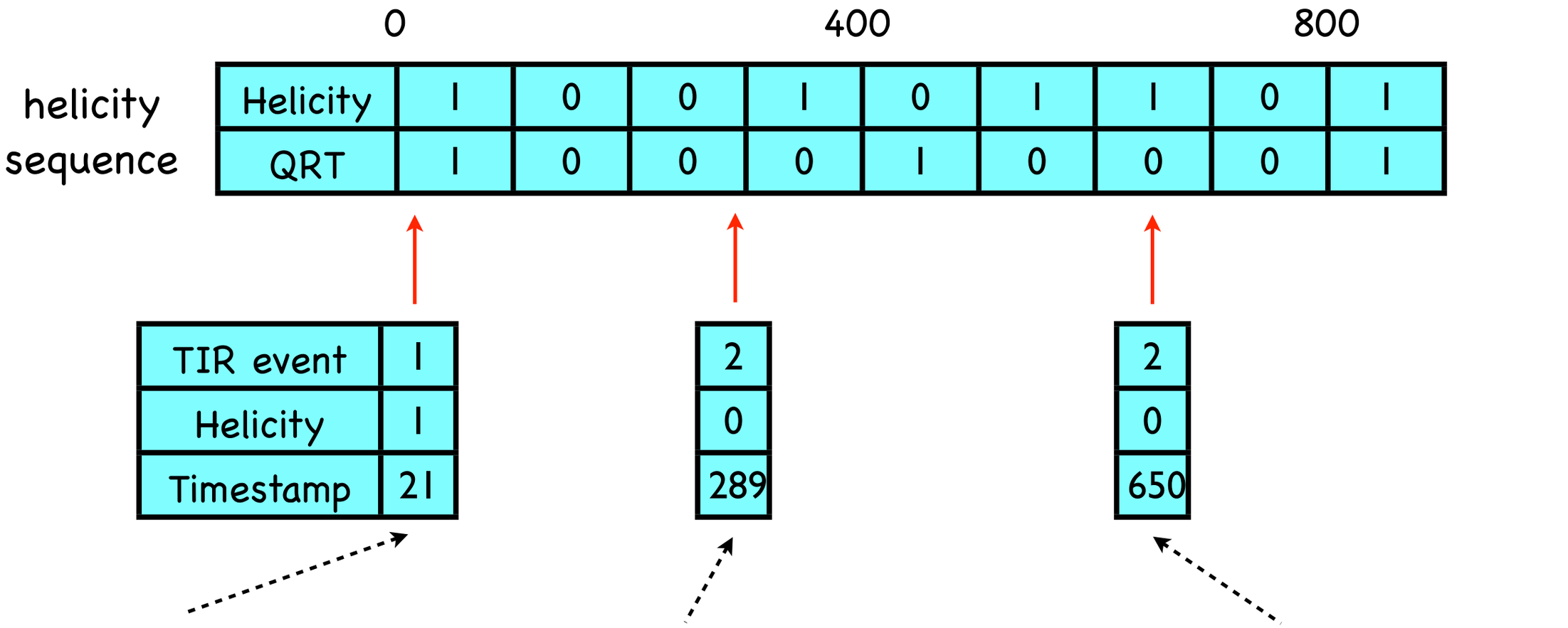
- Solution: Time Stamp
- Use time stamp to decide if there are multiple CODA events in one helicity window or if there are lost windows



Decoder: TIR

- Under real situation, there is no absolute time stamp for the helicity sequence
- The first helicity window in each helicity pattern is the most important for prediction because it decides the polarity
- From now on, we will call a CODA event in the first helicity window of one pattern a "QRT event", and call this window a "QRT window"
- Assume there is already one QRT event, then we can use relative time stamp to decide whether there is a lost QRT window

Decoder: TIR


0
400
800
helicity sequence
Helicity 1 0 0 1 0 1 1 0 1
QRT 1 0 0 0 1 0 0 0 1
TIR event 1
Helicity 1
Timestamp 21
2
0
289
2
0
650
Assume TI recorded a QRT event here
If the next TIR event has a timestamp of 289, the relative timestamp is 289-21=268<400, so this event and the QRT event are in the same pattern, no QRT window is lost
If the next TIR event has a timestamp of 650, the relative timestamp is 629>400, so this event and the QRT event are not in the same pattern, a QRT window is lost

Decoder: TIR

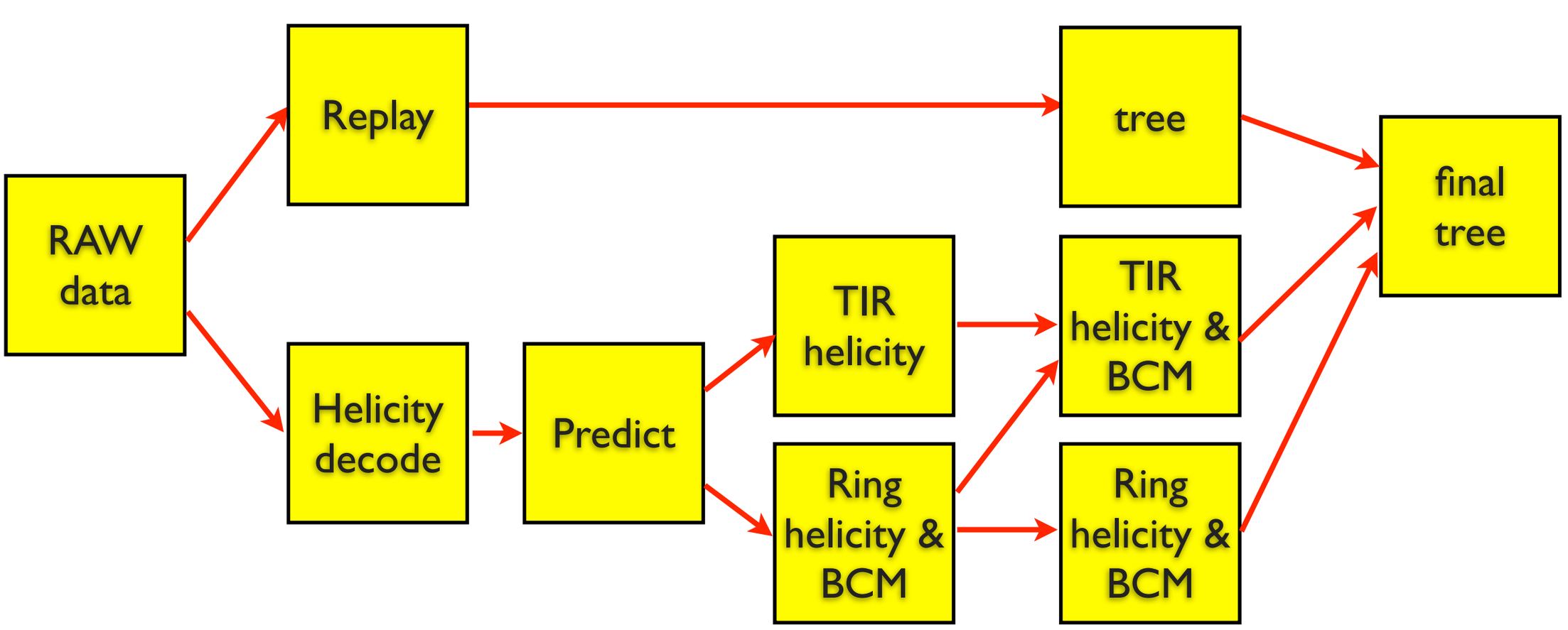
- Use the method described in previous slide, we could decide if there is lost QRT windows
- Once we decide how many QRT windows is lost, we could apply some correction for the prediction to get the correct result
- To decide if there are multiple events inside one helicity window, one could compare the time stamp of 2 events (2 events with the same Helicity, QRT, PairSync value), if the difference is smaller than window time, then these 2 events belong to the same helicity window

Decoder: Backward Prediction

- We also use backward prediction to check the result
- Backward prediction use the same method as forward prediction, but start from the last event
- The algorithm used by prediction changed from
  New Bit1 = Bit30 ^ Bit29 ^ Bit28 ^ Bit7 to
  New Bit30 = Bit30 ^ Bit29 ^ Bit8 ^ Bit1
- After backward prediction, we will recover all of the helicity information used to generate pseudo-random helicity seed

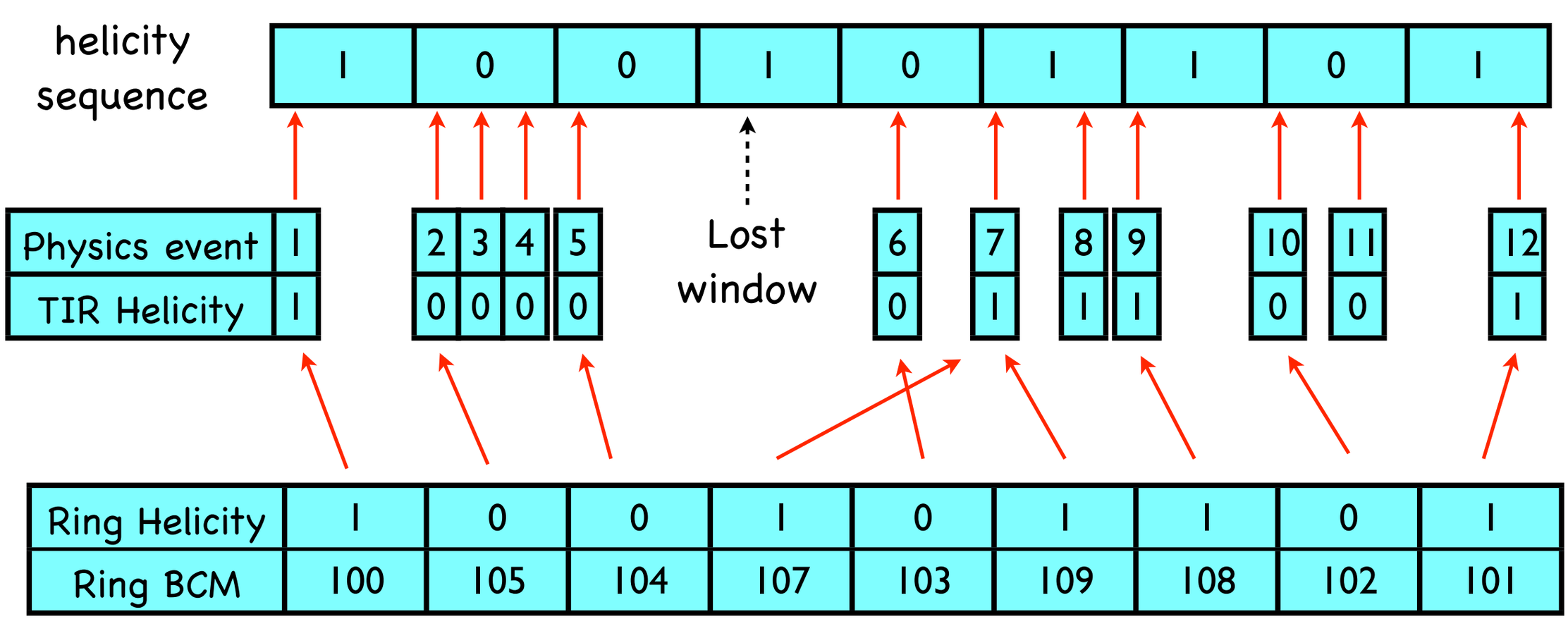
Decoder

- Independent decoder: do not depend on analyzer and replay precess to improve speed
- Since BCM information is recorded independently by scaler, so we need to find a way to align the scaler helicity and TIR helicity

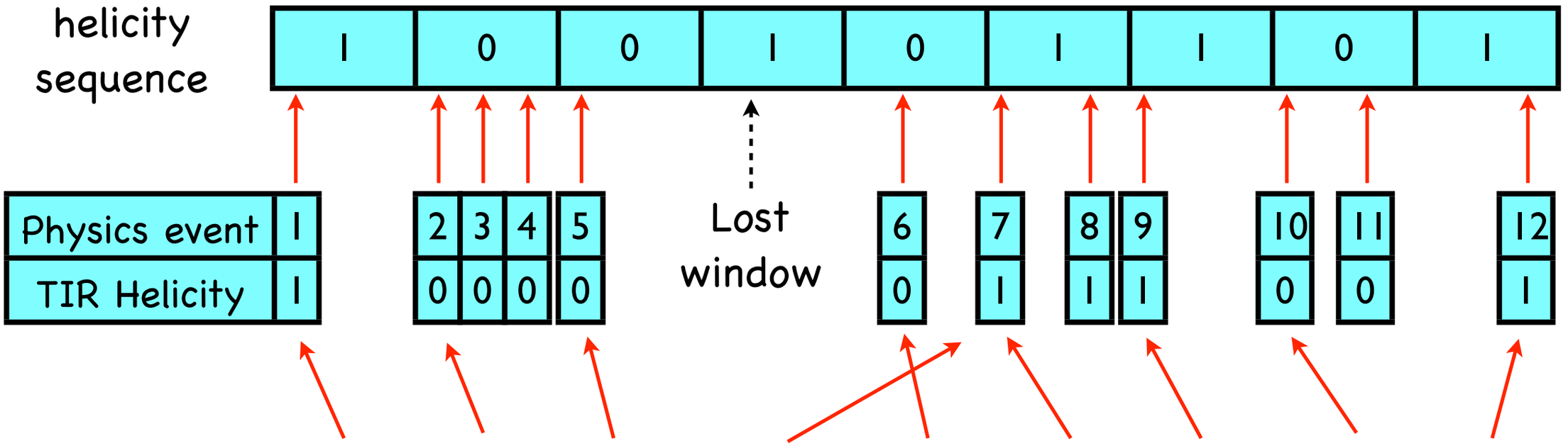


Decoder: Align TIR and Ring

- Use seed of random generator as fingerprint to align TIR and Ring helicity
- For those helicity windows which lost in TIR, we will assign their BCM information to the closest physics event with the same helicity state



Decoder: Align TIR and Ring



Ring Helicity	1	0	0	1	0	1	1	0	1
Ring BCM	100	105	104	107	103	109	108	102	101

Event Number	1	2	3	4	5	6	7	8	9	10	11	12
TIR Helicity	1	0	0	0	0	0	1	1	1	0	0	1
Gated BCM	100	105	0	0	104	103	216	0	108	102	0	101

final event sequence with helicity gated BCM information

Decoder

- The package will provide these tools:
  - extract: decode the helicity information from raw data file
  - ring: predict ring buffer helicity for scaler and HAPPEX DAQ
  - tir: predict TIR helicity in 2 ways (controlled by command line parameter):
    - use seed from ring buffer helicity
    - do not use ring buffer helicity
  - align: assign gated BCM information for each physics event
  - gen: insert the helicity and information to the tree
  - hel: a script which shows how to use these tools

Decoder

- Output of this package is 3 trees:
  - "hel" tree: predicted TIR helicity, gated BCM information from ring and HAPPEX, index to ring and HAPPEX helicity
  - "hel_ring" tree: original data and predicted Ring helicity
  - "hel_happ" tree: original data and predicted HAPPEX helicity

Decoder

- Structure of "hel" tree:
  - hel_rep: raw helicity state
  - hel_act: predicted helicity state
  - qrt, mps, pairsync
  - timestamp: TIR timestamp for each event
  - seed: 30bits pseudo-random helicity seed
  - error: error level of the decoding
  - numring, numhappex: ring buffer event amount recoded in this event
  - bcmup, bcmdown, time: helicity gated information calculated from ring buffer

Decoder

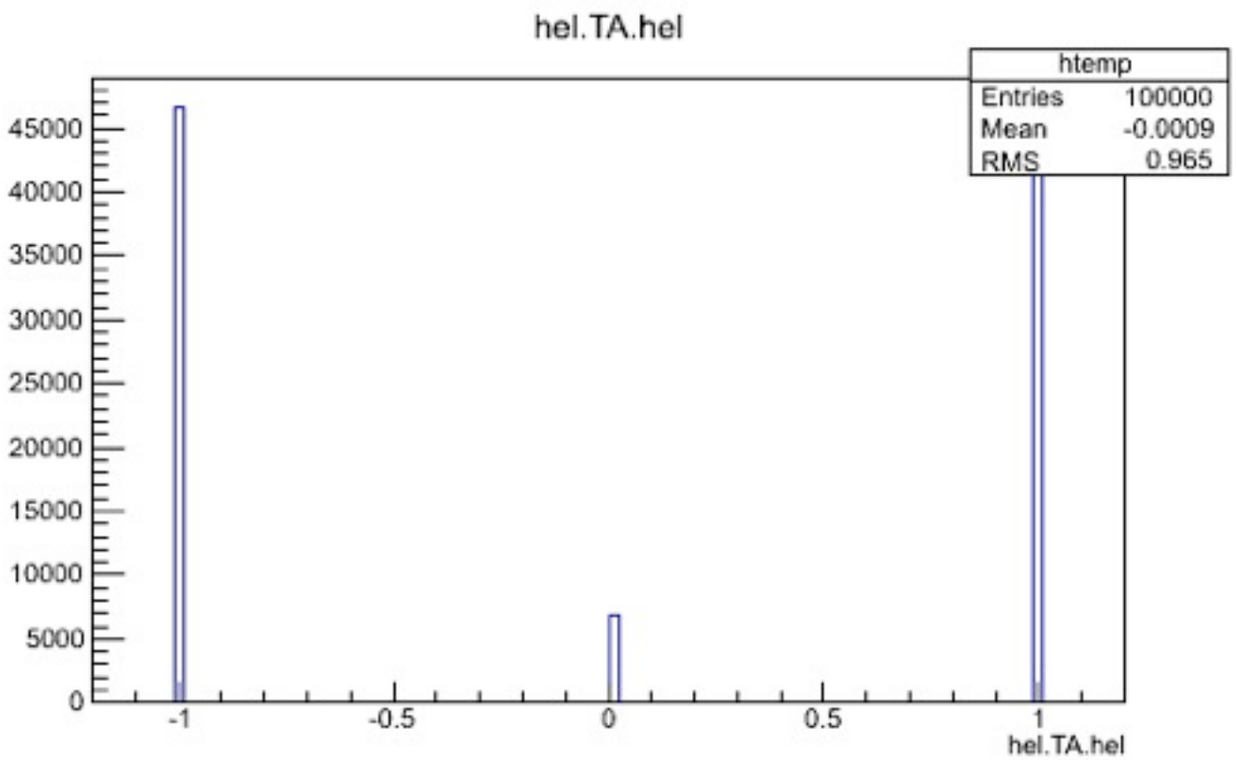
- Structure of "hel_ring" and "hel_happ" tree:
  - evnum: event number which contains this ring buffer event in raw data
  - hel_rep: raw helicity state
  - hel_act: predicted helicity state
  - qrt: QRT state (ring buffer do not have MPS and PairSync)
  - seed: 30bits pseudo-random helicity seed
  - error: error level of the decoding
  - bcmup, bcmdown, time, L1A, T1, T2 ... : helicity gated information recorded ring buffer, different ring buffer will record different information

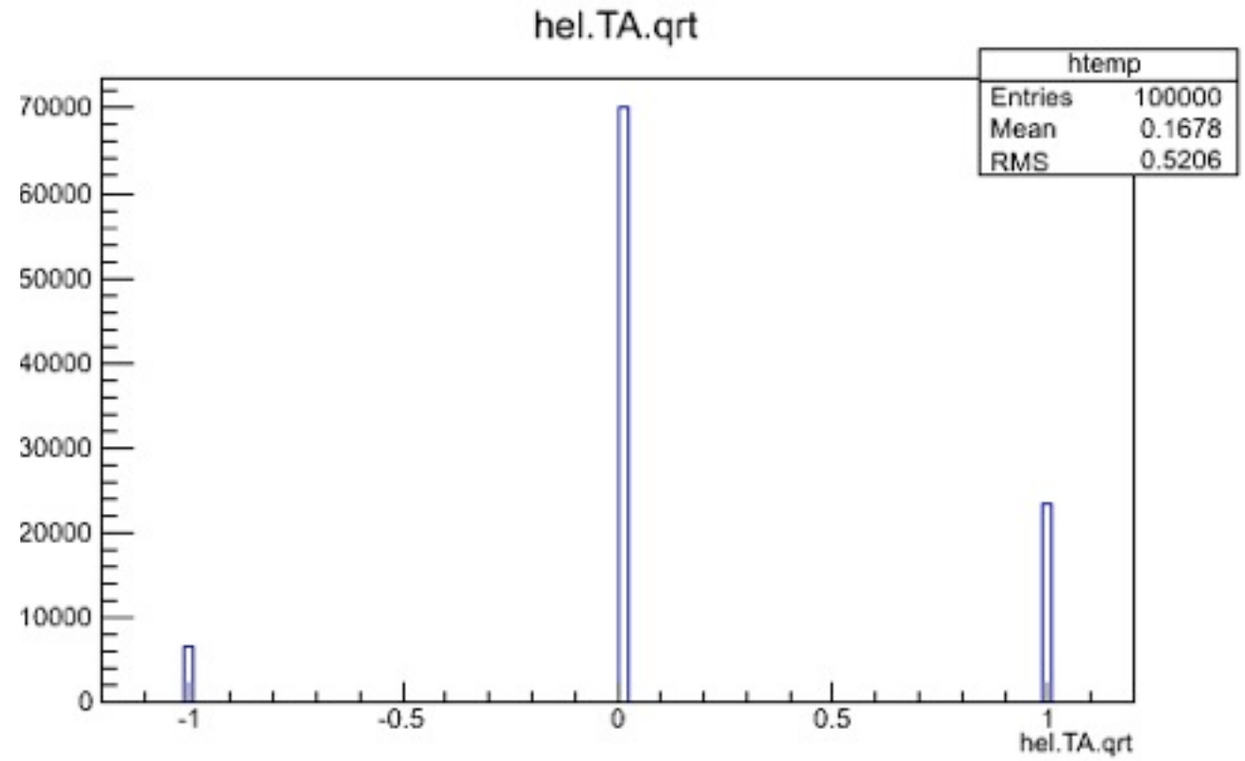
Decoder: Efficiency Test

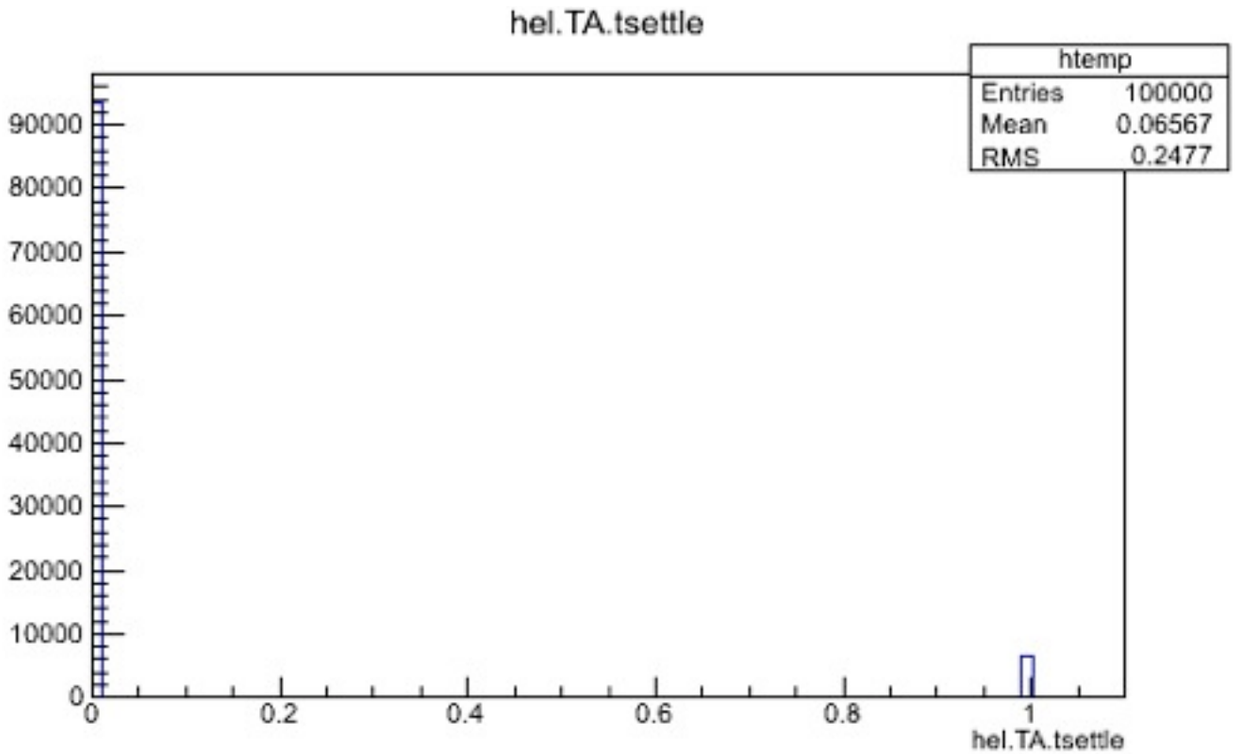
- Test the package with different event rate to see how many event will lose during decode
- The result shows that typically the lost event amount is 0.2-0.5 s times event rate

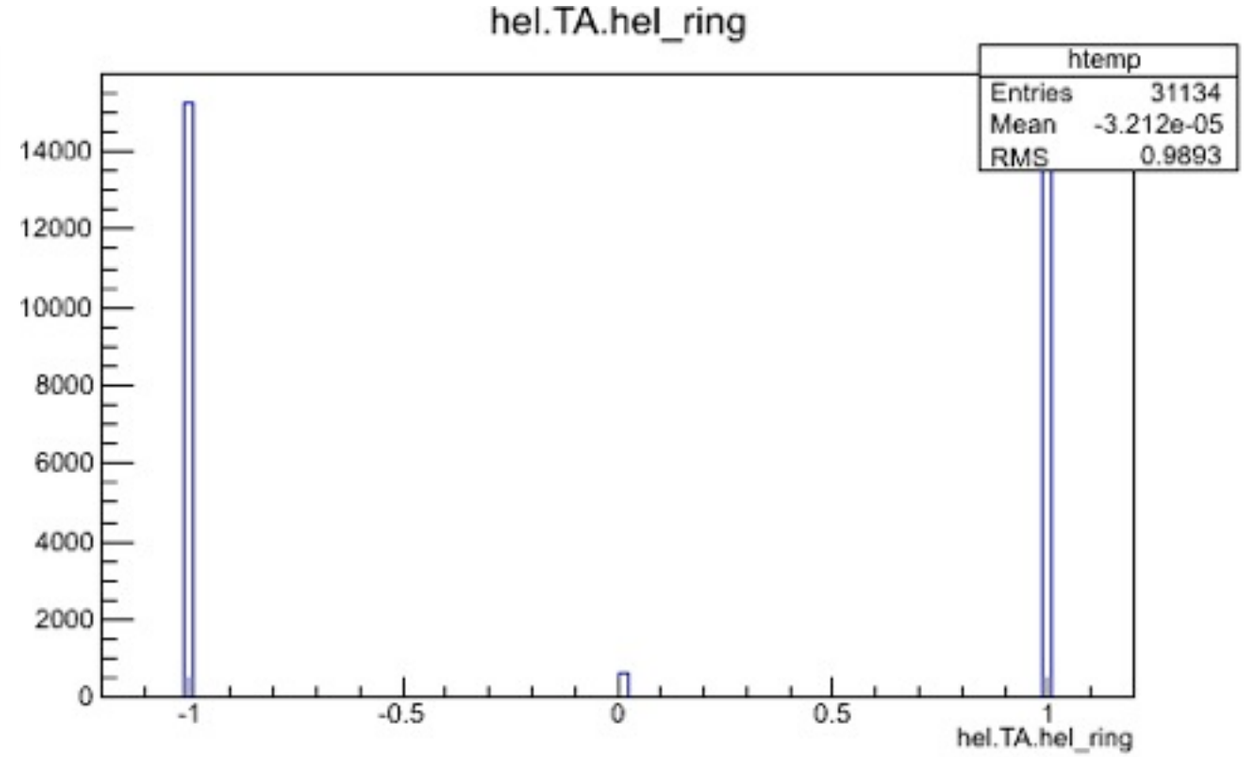
Event Rate/Hz	Event Amount	Event lost for MPS window	Event lost for decode
320	100k	6552	188
1k	100k	6714	153
2k	100k	6709	143
3k	100k	6567	307
5.5k	100k	6715	677

Decoder: Efficiency Test









Charge Asymmetry Test

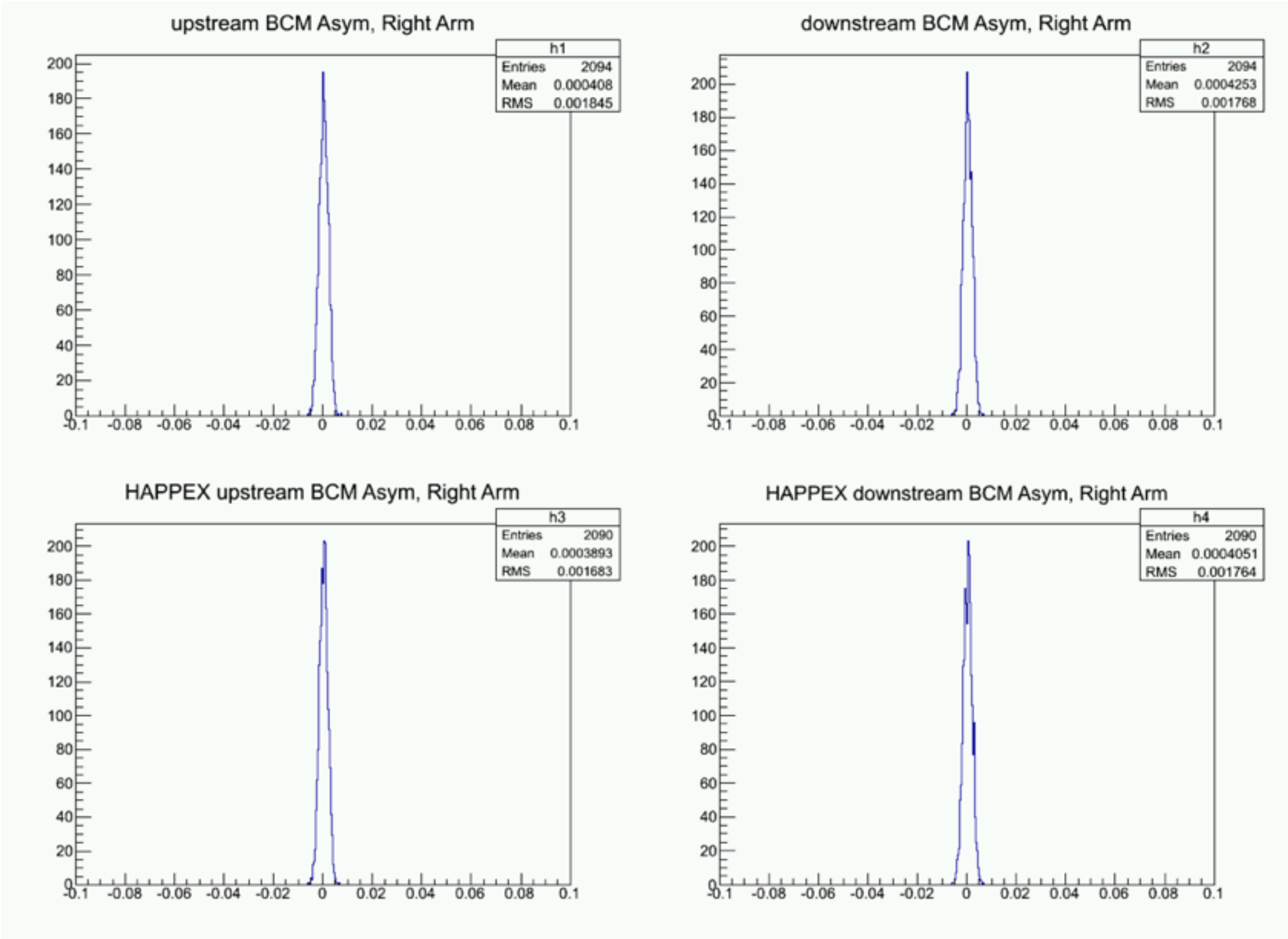
Back to Outline

Charge Asymmetry

- Use charge asymmetry to test the package
- Require large charge asymmetry beam from injector and measure it with HRS DAQ, HAPPEX DAQ, moller DAQ and HallC DAQ
- See Halog 363194 for details

IA-set	LHRS	RHRS	LHAP	RHAP	3rd	Moller	Hall C
0	-0.91%	-0.91%	-0.92%	-0.91%	-0.90%	-0.92%	-0.91%
15000	-0.56%	-0.56%	-0.56%	-0.56%	-0.56%	-0.56%	-0.56%
31000	-0.092%	-0.095%	-0.097%	-0.097%	-0.092%	-0.090%	-0.094%

Charge Asymmetry



Charge Asymmetry: Analysis

- The HRS DAQ, Moller DAQ and HAPPEX DAQ's results looks consistent with each other
- The other purpose of this charge asymmetry is to make sure wether we can reach an statistical uncertainty of <100ppm with an 15 min run
- For run 22194, the event amount is about 10M and it took about 30 min. We use first 5M events (15 min) to make the calculation. We can estimate the statistical uncertainty like this: RMS/Sqrt(entries) = 0.0019/Sqrt(2094) = 42ppm. So it is possible

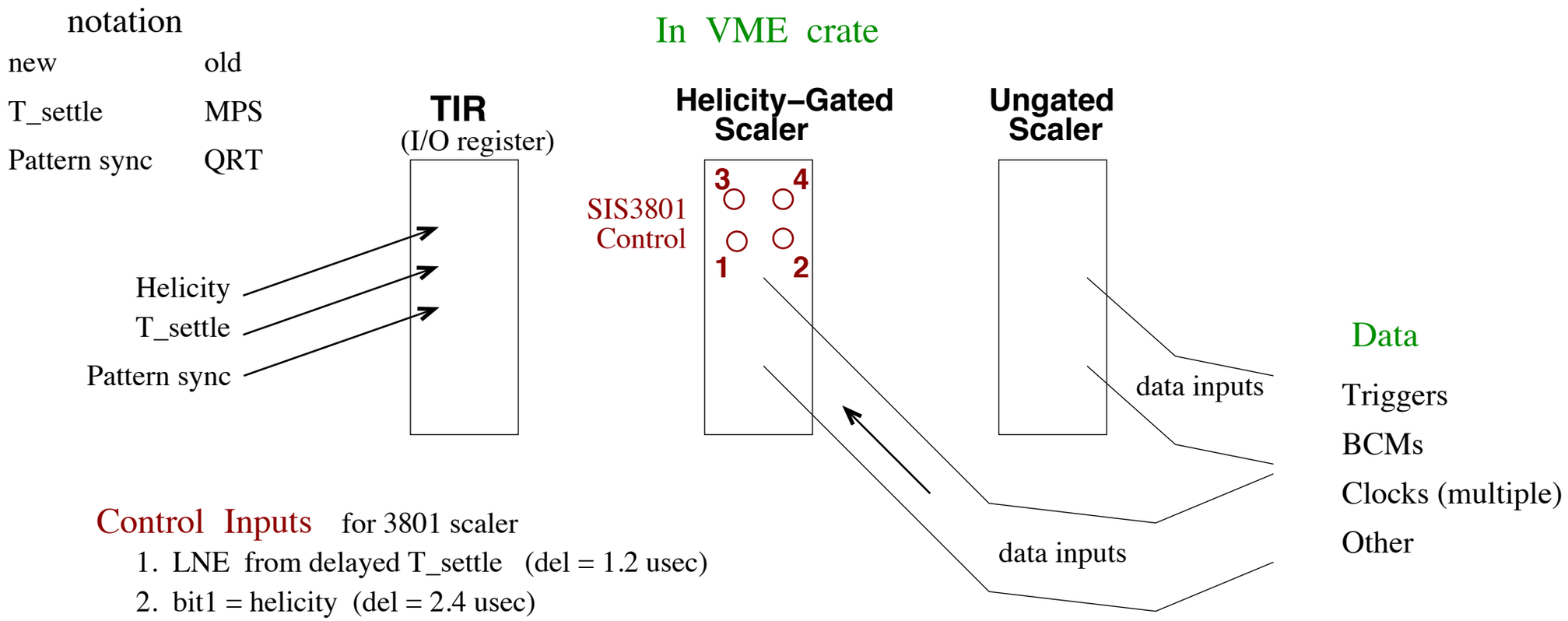
Appendix

Back to Outline

Electronics Scheme for TIR & Scaler

Schematic of Electronics for Helicity Info in Hall A HRS during Qweak

R. Michaels
July 27, 2010



Control Inputs for 3801 scaler

1. LNE from delayed T_settle (del = 1.2 usec)
2. bit1 = helicity (del = 2.4 usec)
3. bit2 = pattern sync (del = 2.4 usec)
4. Gate = T_settle with appro. polarity and undelayed. (TRUE disables counting)

At frequency "f" (f = 1 kHz normally) we must:

1. Read helicity–gated scaler with zero deadtime.
2. Keep a queue in memory of the cpu of [helicity, pattern sync, and clock(s)]
3. Sort the plus and minus helicity, adding the data to "virtual" scalers for online consumption.
4. Check data for errors. Issue warnings.

For each trigger accepted by Trigger Supervisor, we must:

1. Read the (a) TIR bits and (b) clocks from ungated scaler.
2. Flush the memory queue (see 2 above), to have full sequence of helicity data.

Back to Slide