New Check-Sort-Push protocol in iRPC data compression, transmission and decompression in Backend electronics system

Zhen-An Liu for CMS Muon Group

RPC2022, CERN Geneva

Sep. 26-30 2022

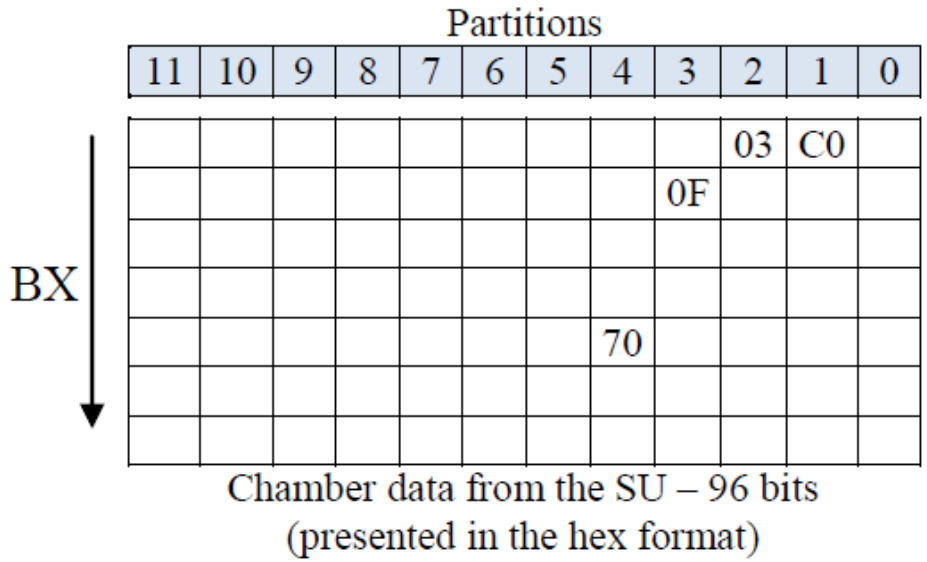
Outline

- Data transmission of present RPCs
- Why do we propose Check-Sort-Push in iRPC
  - More data to be transmitted(Timing data)
  - Still low occupancy
  - Receiver Latency Window
  - Position priority problem with unpaired channels
  - Time based priority
- Implementation and test results
- Summary

Data transmission of the present RPCs

- Data compression is mandatory for Modern experiments like CMS
- Preset CMS RPC defines
  - Partitions: hits in 8 strip with HEX
  - Partition number 0:11
- LinkSystem: Compression by Zero-Suppression and transmission
  - Data production in BX(25ns)
  - Transmit only partition with Non-zerc partition data (Multiplexing)
    - Data Structure Part.Num+Delay+Part.dat
    - Pipelined with delayed BXs(25ns)
  - Maximum delay is 8 BX(25ns)

- CPPF/OMTF (Concentration Processing and Fanout / Overlap Muon Track Finder)
  - Reception/DeMultiplexing
  - Clusterization



LinkBoard System(sender)

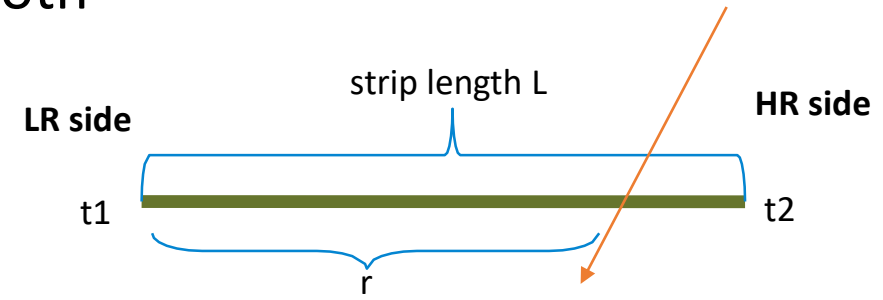
EOD	Partition delay	Partition number	Partition data
0	0	2	03
0	1	1	C0
0	1	3	0F
0	0	4	70

Coder output data

To CPPF System(receiver)

iRPC FEE-BEE Emulation

- iRPC provides better position by timing measurement from both strip ends
- Frontend Electronics board (FEB) emulator
  - Hit position
    - *r* is calculated by the time difference of signals from both Ends(see next page).
  - Digitization
    - For each fired strip, there are always 2 32-bit TDC data constructed.
      - Channel-HR Rising Edge + Channel-LR Rising Edge
  - Zero-suppression
    - Sends time info from only fired strips
  - TDC Data format(32 bits)
    - devAddr： FPGA ID
    - chanAddr : channel address
    - Coarse time: combine BCN(Preserved, 12 bits) and t1, t2
    - Fine time: responsible to the precision, 2.5ns/256≈10ps



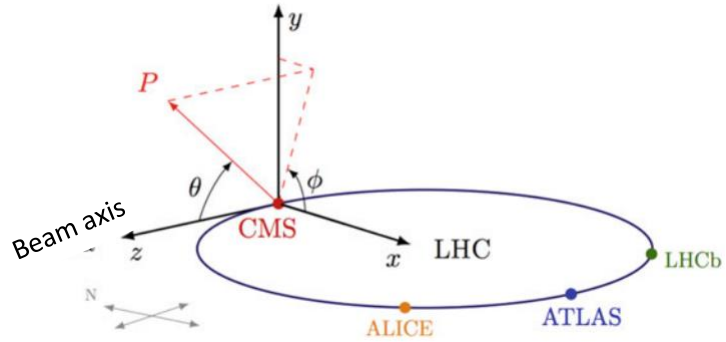
$$r = \frac{1}{2}\mathrm{L} - \frac{(\mathrm{t2} - \mathrm{t1})}{2} * v$$

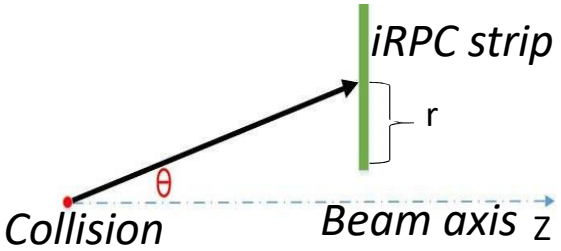
devAddr	chanAddr	TDC data	
		Coarse time	Fine time
2	6	16	8

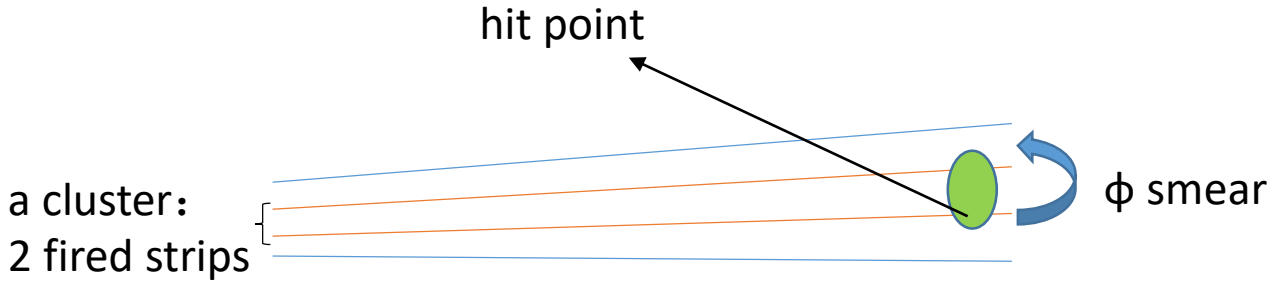
Transmit(Tx) Latency and Receive(Rx) Latency are introduced for MuX/DeMux

iRPC data source from simulation

- Data generation
  - Random hit point(r, ϕ) in a chamber
    - r: generates the hit point along strip.
    - ϕ: generates the strip number.
  - Smear of ϕ direction :
    - Generates a consecutive set of strips as a cluster.
  - Cluster size randomization :
    - 1-8,mean is 2.35
  - Number of clusters :
    - 1(75% probability),2(25% probability， keep the cluster size as 1)







Latency Window optimization

- Efficiency with different Tx/Rx Latency(Maximum Delay) Window
  - TxLW settings(BX): 12, 18, 24
  - Average hit rate(Hz/cm^2 ,*safety factor 3*):600, 800, 1000, 1200, 1500, 1800, 2000
  - Mean cluster size:2.35 ; Surface of half chamber :6600cm^2 (1 fiber)
  - Size of the information/channel : 32bit (only rising edge)

- For maximum rate 2000 Hz/cm^2,data rate
  = 2000_{rate}*$6600_{surface}$* $2.35_{cluster\ size}$ *$64_{1\ strip\ info}$ Gb/s

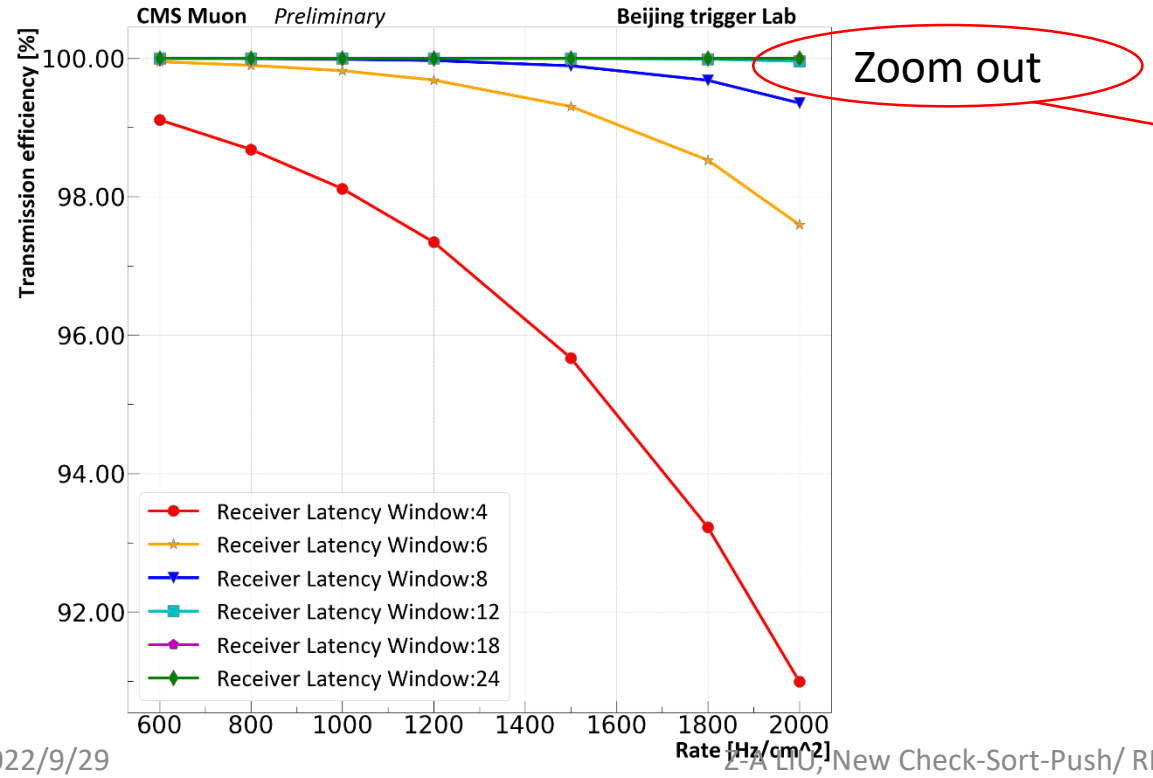
thus data rate is 1.99 Gb/s(smaller than GBT data bandwidth)

- A larger TxLW leads to increased efficiency.
- 100 % efficiency is measured with a TxLW setting of

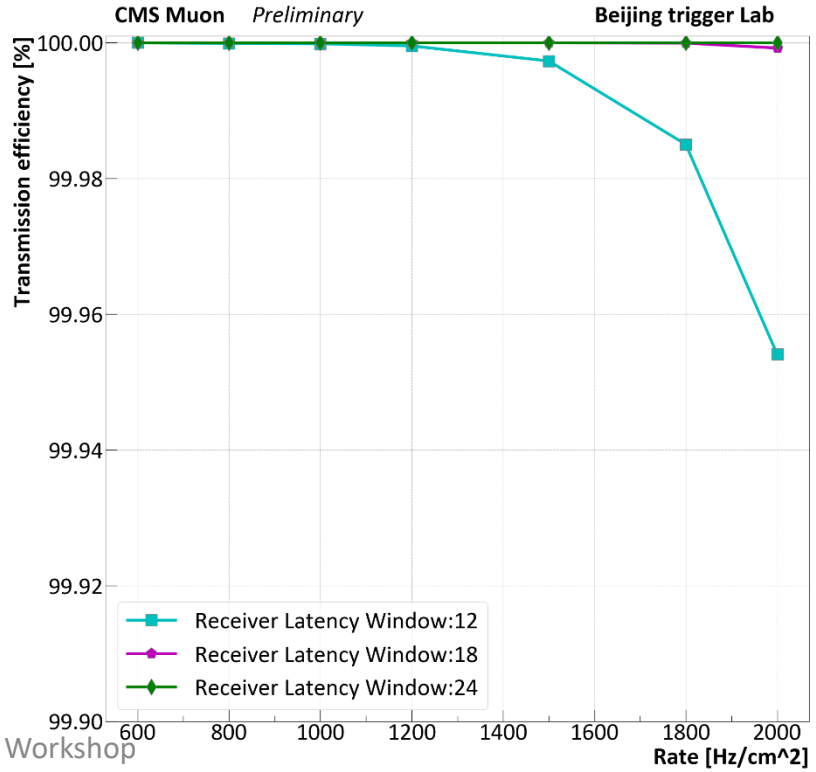
24 BX in our simulation.

Emulation Study for high occupancy case

- Emulation Study results with real BEE Board (BEB)
  - FEE sends all data(TxLW: 24)
  - BEE introduce Receive Latency Window(RxLW)
    - The transmission window is calculated by comparing sending BX and the BX the data originates. At high occupancy cases when the data to be transmitted more than the transmission window should be rejected and sending a truncate flag to the backend.



Zoom out



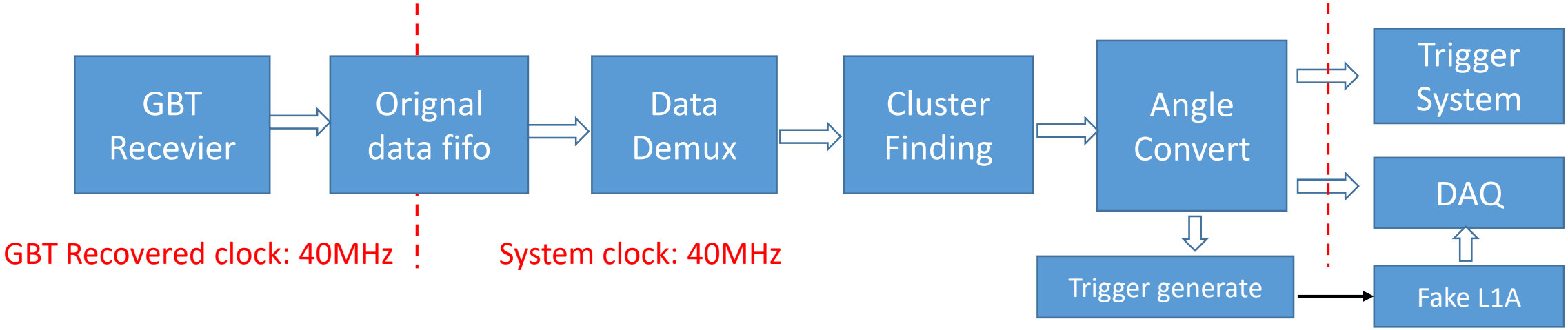
FEB sending priority

- A new proposal ("Check-Sort-Push" algorithm based on timing) was suggested to have shorter Latency Window so to ease backend electronics design such a way that
  - Check for new hits every BX(25ns) ,
  - Sort the new hits with existing ones in sequence of production in transmission buffer,
  - Push the earliest hits in the buffer for transmission.

BEE firmware algorithm

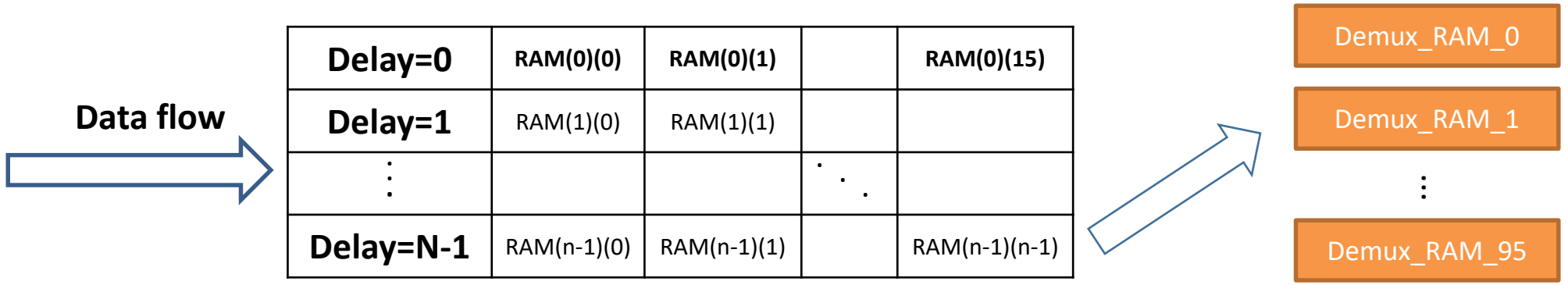
- Key point of the algorithm
  - Use pipeline method
  - Data decompression
  - DAQ module packs and uploads the original data and angle information based on L1A arriving time

after latency adjustment.



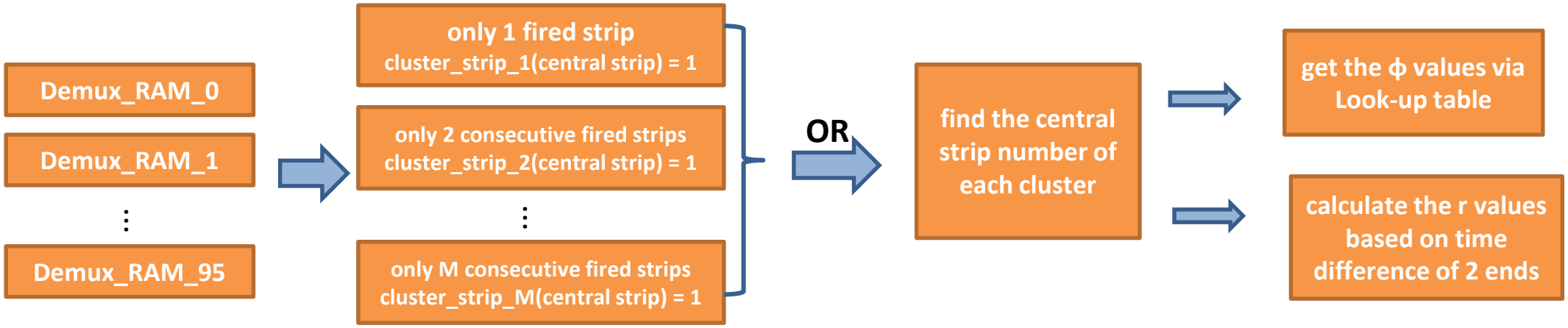
BEE firmware algorithm

- Data De-multiplexing



- Cluster Finding

Angle Convert



Problems spotted in Cosmic-Ray Test

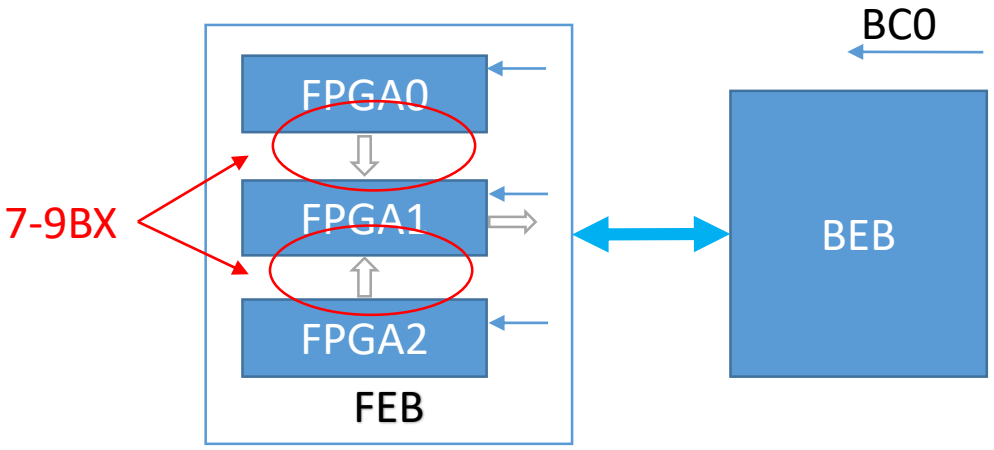
- Joint Test of iRPC + FEE + BEE full system at CERN 904
- Analysis of Cosmic-ray test data showed

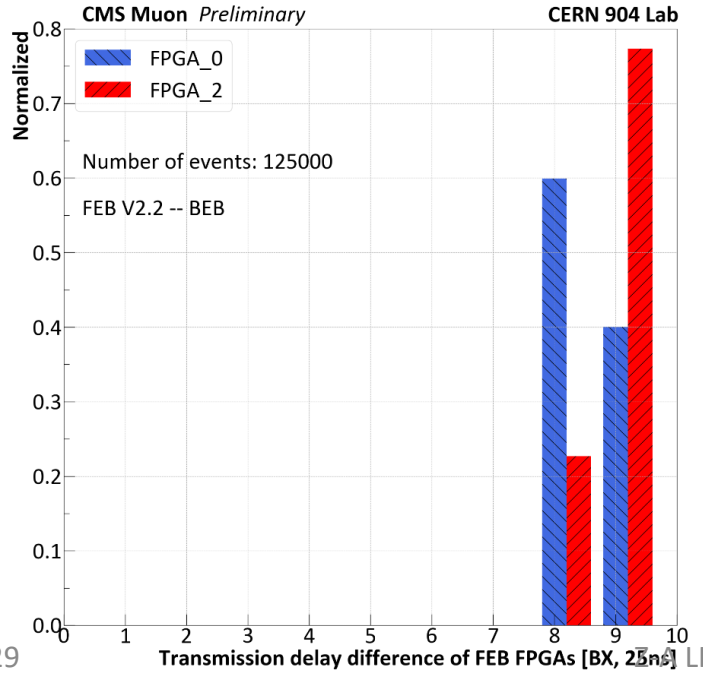
too many fake/false data due to

  - More hit data from only one end
  - Data produced at same BX received in too far separated (~20bx)

- Careful Study found that
  1. there is a strong dependence of data with the strips and
  2. unfixed separation.

Problems spotted in Cosmic-Ray Test(2)

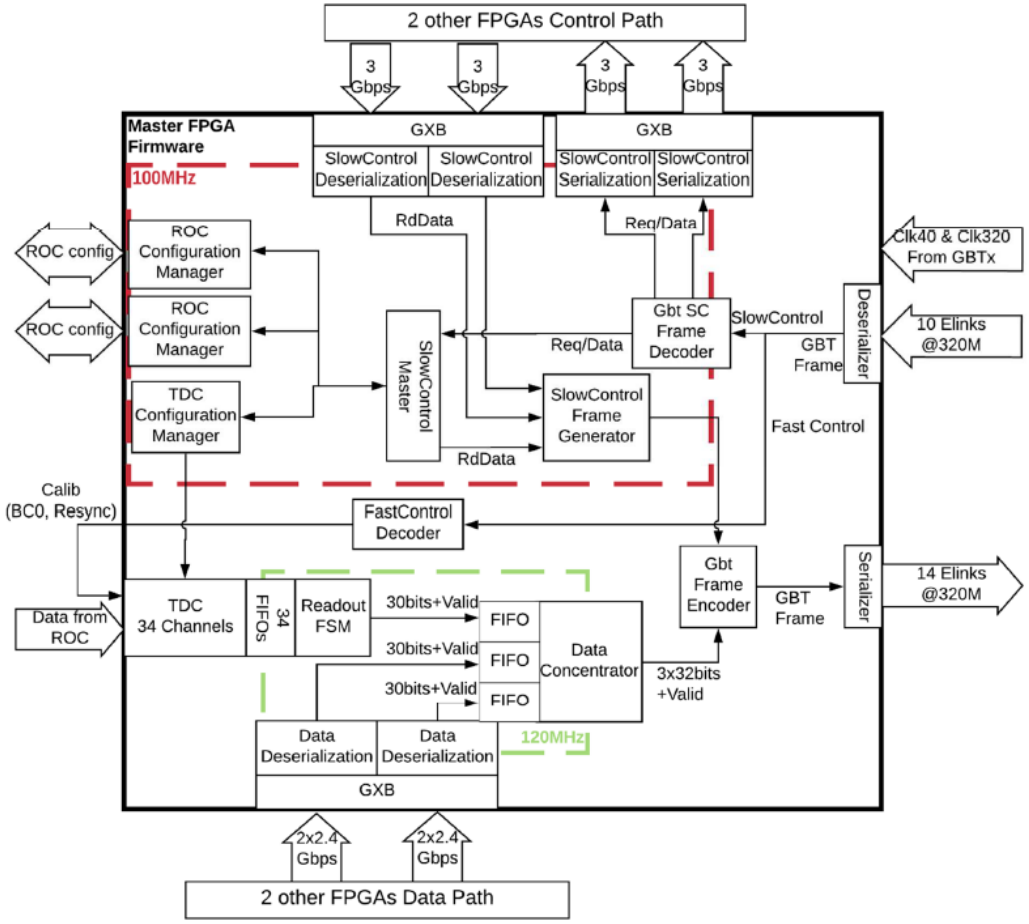




- Transmission delay difference
  - The backend sends the resync signal to the FEB and the FEB will reply with its timestamp
  - The TDC data of the FEB FPGA0/2 are found with a delay compared to FPGA1.
  - The delay is not fixed and measured to be 7-9 BX.
    - **BX** : Bunch crossing, 25ns per BX.
  - The distribution changes after each powering up.

TDC/FPGA priority was used in FEB

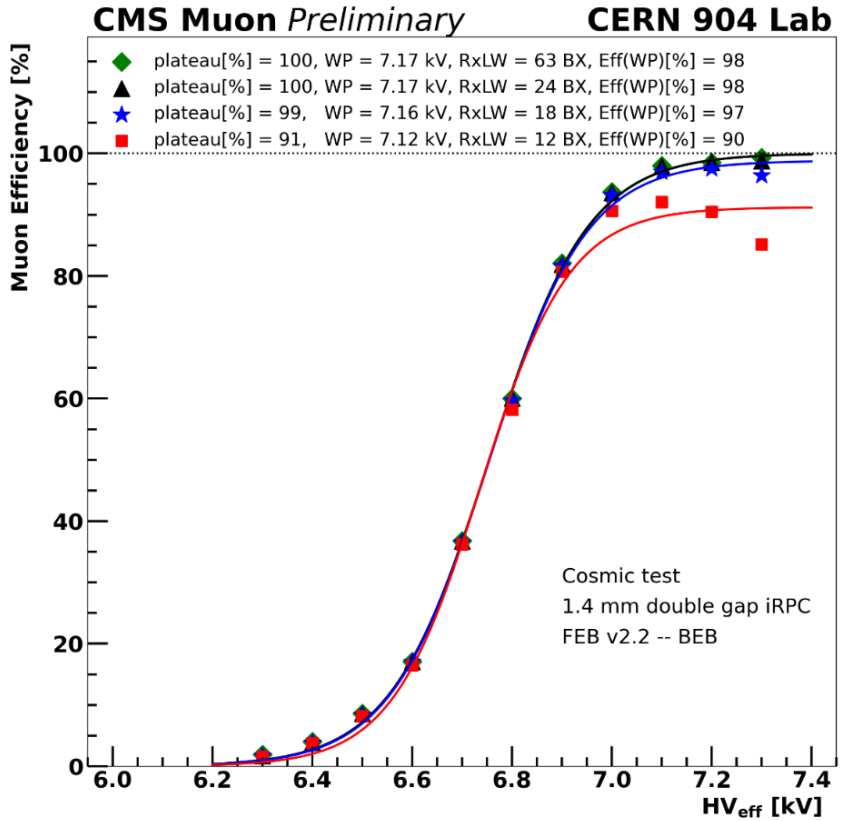
- Joint work FEE +BEE team concluded that
  1. Fixed FPGA and Channel sending sequence is used(Check-Sort-Push was not implemented in FEE yet), and
  2. strips are grouped into three TDC-FPGAs, which are in fixed priority(read order - Channel ID and FPGA ID), and
  3. data of one strip from both ends are not in the same GBT frame hence not with same priority/order



FEB specification v 0.7

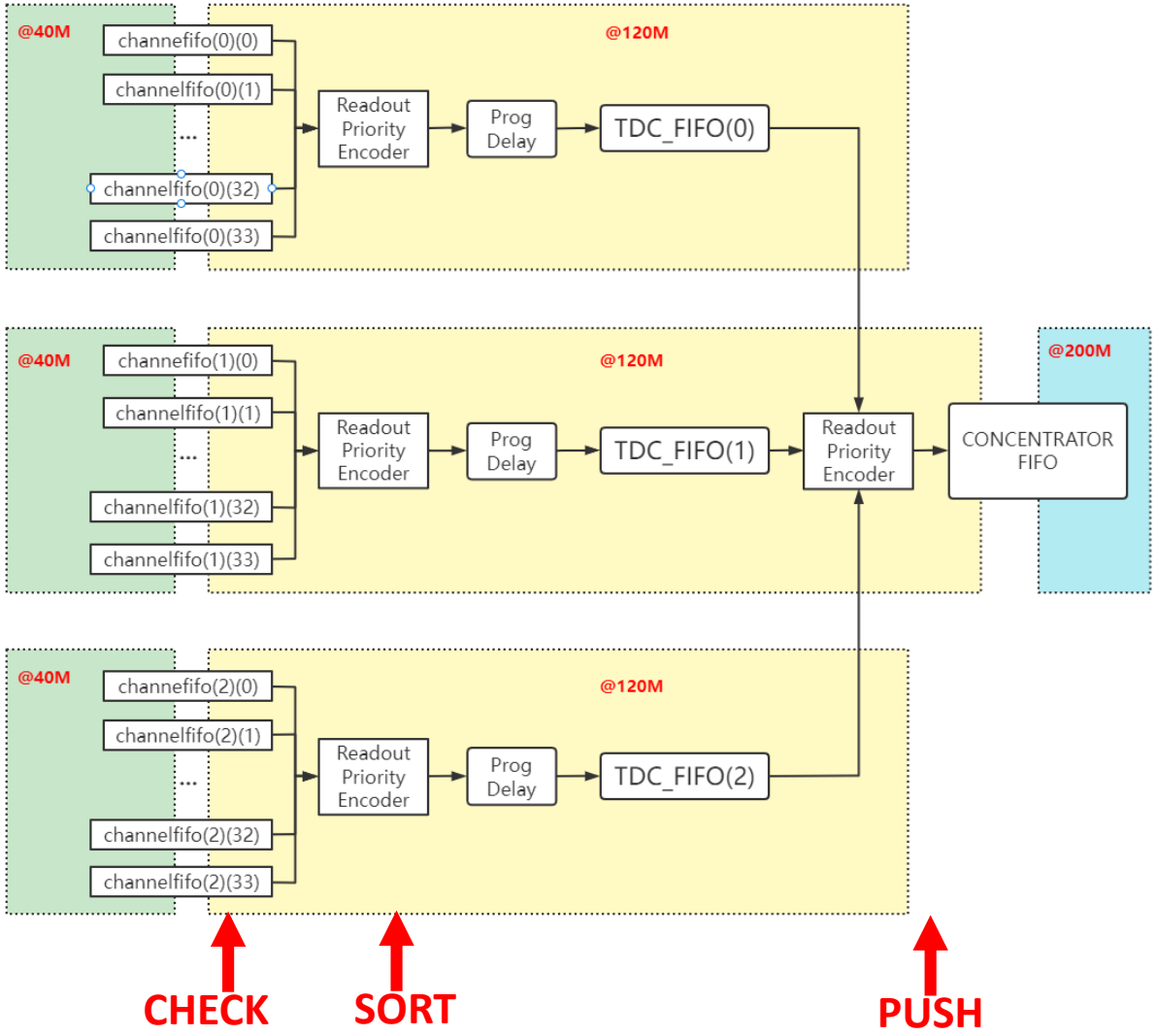
FEB-BEB joint test --Efficiency scan

- This result was acquired using the present FEE parameters and backend system in 904.
  - When RxLW = 12 BX(25ns), the efficiency was not high.
  - And it increases to 100% when the RxLW set as 63 BX(25ns) (Extreme test situation, at the cost of 1 link per BEB).
- Shouldl be improved by implementing Check-Sort-Push mechanism in FEB.



Check-Sort-Push is mandatory

- Agreement has been made with FEE team that time priority(sorting) will be introduced and Check-Sort-Push protocol will be used in FEE in next version
  - Sorting based on time stamp in FPGA separately and then
  - Merged before sending



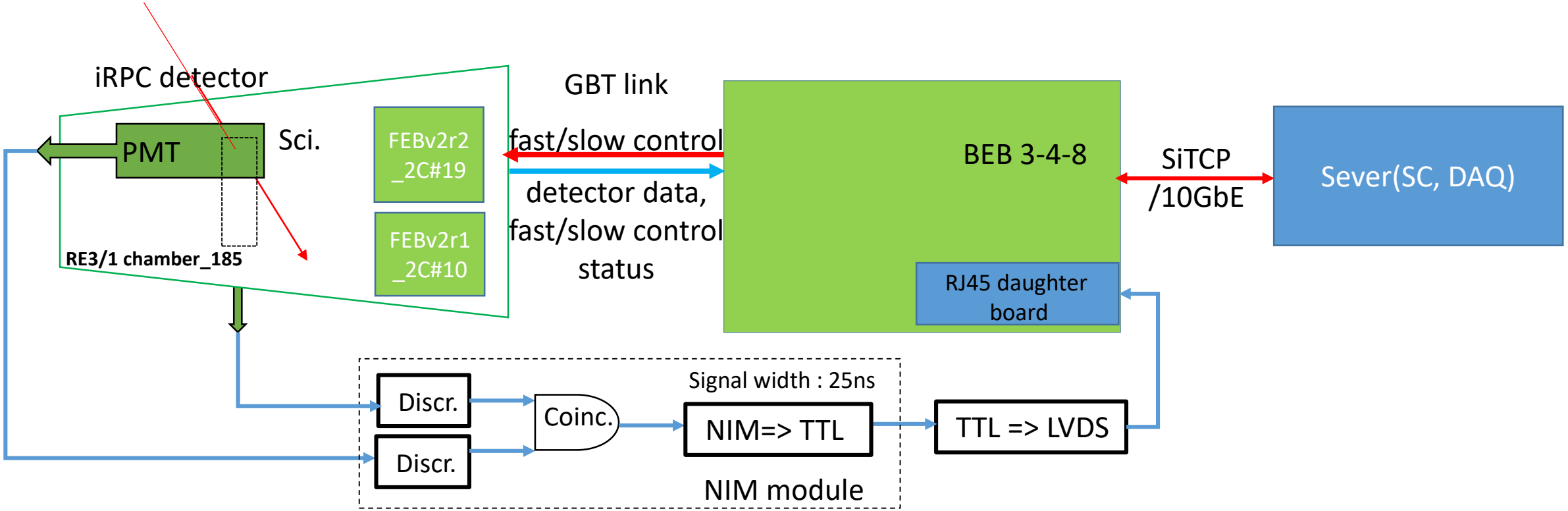
Summary

- Check-Sort-Push Protocol was proposed in the year 2020 for CMS iRPC/RPC data transmission
  - Both Simulation and Emulation showed that such a mechanism is needed
- Cosmic-Ray data taking and analysis shows that this Protocol is mandatory
- It’s Implementation is under taking in iRPC/FEE

backup

- Purpose
  - Studying the data transmission mechanism between the front and backend.
  - Verifying the backend functions(fast/slow control, data transmission mechanism and DAQ data format, etc.)

- System setup



Strange Data reception

Treceive	TDCdata1	TDCdata2	TDCdata3	converted_each_data_to_strip_and_time(ns)		
BX-4	464002d3	454002f2	463ff459	strip_22_HR :40967.1	strip_21_HR :40967.4	strip_22_HR :40930.9
BX-3	474002c8	483ff1f2	473ff29c	strip_23_HR :40967.0	strip_24_HR :40924.9	strip_23_HR :40926.5
BX-2	4b3ff25d	4a3ff245	493ff22e	strip_27_HR :40925.9	strip_26_HR :40925.7	strip_25_HR :40925.4
BX-1	4e3ff426	4d3ff361	4c3ff289	strip_30_HR :40930.4	strip_29_HR :40928.4	strip_28_HR :40926.3
BX+0	003ff73d	4e4002b8	4d4002b0	strip_32_HR :40938.1	strip_30_HR :40966.8	strip_29_HR :40966.7
BX+1	033ff71b	023ff744	013ff74a	strip_35_HR :40937.8	strip_34_HR :40938.2	strip_33_HR :40938.2
BX+2	034002e2	0240030d	0140030e	strip_35_HR :40967.2	strip_34_HR :40967.6	strip_33_HR :40967.6
BX+3	043ff749	01400ed2	00400eae	strip_36_HR :40938.2	strip_33_HR :40997.1	strip_32_HR :40996.7
BX+4	0440030d	053ff767	4f3ff474	strip_36_HR :40967.6	strip_37_HR :40938.5	strip_31_HR :40931.1
BX+5	073ff786	063ff767	054003eb	strip_39_HR :40938.8	strip_38_HR :40938.5	strip_37_HR :40969.8
BX+6	083ff78e	074003b6	0640031d	strip_40_HR :40938.9	strip_39_HR :40969.3	strip_38_HR :40967.8
BX+7	094003c0	084003b7	093ff79f	strip_41_HR :40969.4	strip_40_HR :40969.3	strip_41_HR :40939.1
BX+8	0a4003c7	0b3ff7b1	0a3ff7b4	strip_42_HR :40969.4	strip_43_HR :40939.2	strip_42_HR :40939.3
BX+9	0d3ff7be	0c3ff7b3	0b40036c	strip_45_HR :40939.4	strip_44_HR :40939.2	strip_43_HR :40968.6
BX+10	0e3ff7b3	0d4003a6	0c400380	strip_46_HR :40939.2	strip_45_HR :40969.1	strip_44_HR :40968.8
				----------BEB ±10BX window cut----------		
BX+11	0f4004d1	0e4003d6	0f3ff8c2	strip_47_HR :40972.0	strip_46_HR :40969.6	strip_47_HR :40941.9
BX+12	123ffb14	113ffb26	103ffc29	strip_45_LR :40947.7	strip_46_LR :40947.9	strip_47_LR :40950.4
BX+13	143ffafa	133ffb07	114006d0	strip_43_LR :40947.4	strip_44_LR :40947.6	strip_46_LR :40977.0
BX+14	173ffaa6	163ffa30	153ffaf8	strip_40_LR :40946.6	strip_41_LR :40945.5	strip_42_LR :40947.4
BX+15	1a3ffa7c	193ffc4d	183ffa9b	strip_37_LR :40946.2	strip_38_LR :40950.8	strip_39_LR :40946.5
BX+16	1d3ffc25	1c3ffc37	1b3ffae3	strip_34_LR :40950.4	strip_35_LR :40950.5	strip_36_LR : 40947.2
BX+17	503ff17c	1f3ffc15	1e3ffc32	strip_31_LR :40923.7	strip_32_LR :40950.2	strip_33_LR :40950.5
BX+18	523ff067	513ff07b	4f4002e3	strip_29_LR :40921.0	strip_30_LR :40921.2	strip_16_LR :40967.2
BX+19	533ff028	523ffe8d	5140079c	strip_28_LR :40920.4	strip_29_LR :40956.4	strip_30_LR :40979.0
BX+20	563fefb4	553fefb1	543fefc3	strip_25_LR :40919.3	strip_26_LR :40919.2	strip_27_LR :40919.4
BX+21	593ff044	583ff004	573fefdb	strip_22_LR :40920.7	strip_23_LR :40920.0	strip_24_LR :40919.6
BX+22	5a3ffe73	5b3ff048	5a3ff049	strip_21_LR :40956.1	strip_20_LR : 40920.7	strip_21_LR :40920.7

I.FPGA1

II.FPGA0

III.FPGA1

firstly

finally

32bit received GBT frame from FEB in time seq.

- Problem: Data produced in same time in different channel are transmitted with unexpected delay.
- For example Strip 23 HR was transmitted at BX-2 but LR at BX+21.

- Present FEB sending algorithm:
- Data with smaller FPGA ID and channel number are transmitted firstly.

Eg. 2bit FPGA ID+6bit channel

I. “4x” FPGA 1

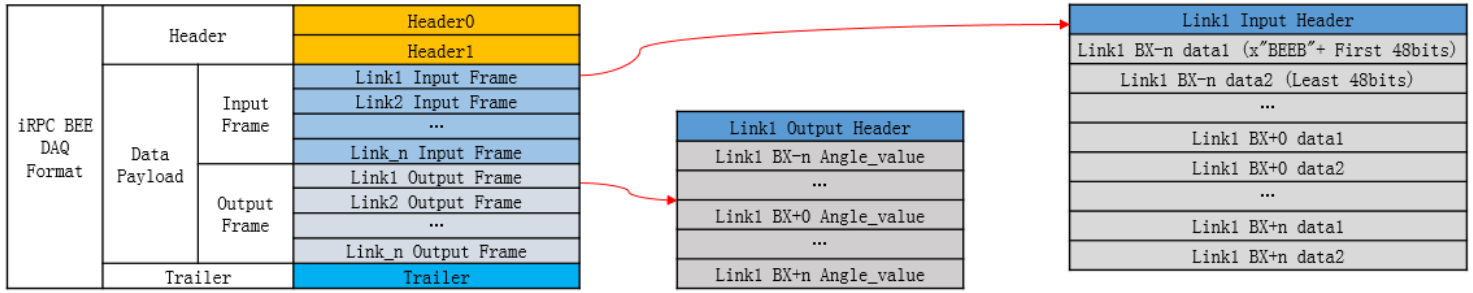
II. “0x-1x” FPGA 0

III. “5x” FPGA 1

Channel number is used in each sector I/II/III.

BEE-DAQ data format

- Current iRPC BEE DAQ data format：
  - Use different identifiers as Header0/1 and trailer.
  - Pack angle value and original data by link number for events generated in the same BX.
  - Not as complete as the CPPF data format for the time being, still working on it!



	Byte 7								Byte 6								Byte 5								Byte 4								Byte 3								Byte 2								Byte 1								Byte 0							
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Header0	Special Mark (x" DEADBEEF")																																Trigger Number																															
Header1	Orbit Number																								Input Windows								Output Windows								Evt_type								Event Length															
Input Link Header	LinkType (x" FA")								Link8 Length (7b)								Link7 Length (7b)								Link6 Length (7b)								Link5 Length (7b)								Link4 Length (7b)								Link3 Length (7b)								Link2 Length (7b)								Link1 Length (7b)							
Output Link Header	Link Type (x" FFFF")																Link8 Length (6b)								Link7 Length (6b)								Link6 Length (6b)								Link5 Length (6b)								Link4 Length (6b)								Link3 Length (6b)								Link2 Length (6b)								Link1 Length (6b)							
Input Link Data	Uplink frame between the FEB and the BEB																																																																						
Output Link Angle_value	1st cluster Theta (R value)																1st cluster Phi																2nd cluster Theta (R value)																2nd cluster Phi															
Trailer0	FSM Error								Link8 Error								Link7 Error								Link6 Error								Link5 Error								Link4 Error								Link3 Error								Link2 Error								Link1 Error							
Trailer1	Board_ID																BEE_version																CRC-32																												Event Length							