(仮題)J-PARC E16 実験の計測システムの現状と課題

高橋智則

計測システム研究会 2021 @ 九州大学 2021.10.28–29

理研仁科センター

内容

- FairMQ, Redis を使った DAQ (五十嵐さんのトークの続き)
- J-PARC E16 実験の概要
- 現状の問題点
- Upgrade plan

高橋の私見が多いです

FairMQ

大規模データ処理のワークフローを実装するためのソフトウェアフレームワーク

- データ処理: シミュレーション, データ解析 (オフライン, オンライン), DAQ
- 非同期メッセージ転送. さまざまな転送方法を提供. (**ZeroMQ**, shared memory, OFI(InfiniBand etc.))
- 転送するメッセージ (データ) の中身はユーザーが自由に定義
- `https://github.com/FairRootGroup/FairMQ`

主に原子核実験で使用されている

- **ALICE** (CERN, 重イオン衝突, QGP)
- **CBM** (GSI/FAIR, 固定標的の重イオン衝突, 高密度 QCD)
- PANDA (GSI/FAIR, 反陽子ビームを使ったハドロン研究)
- etc.

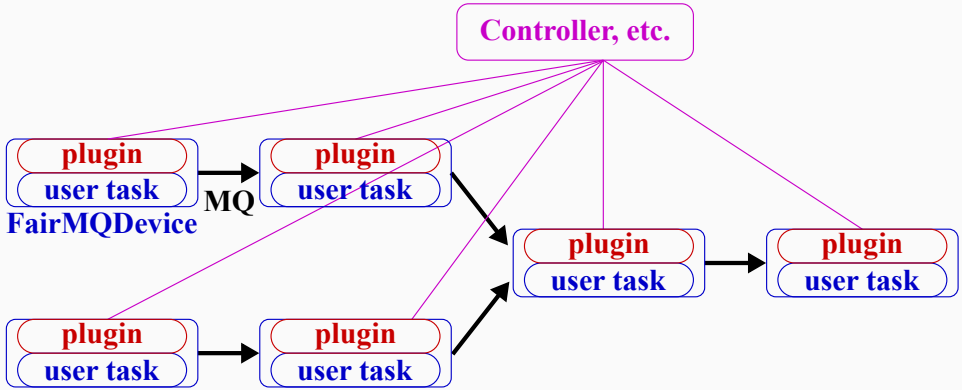
ソフトウェア構成

FairMQ が提供するもの

- ユーザータスクを実行する **Device (+ state machine)**
- Device(~process) 間通信のための **MQ**
- 外部ツール (UI, database) と連携するための **plugin**

用途に合わせてソースコードを書く

複数 device を同時に制御する controller (operator) は別パッケージ



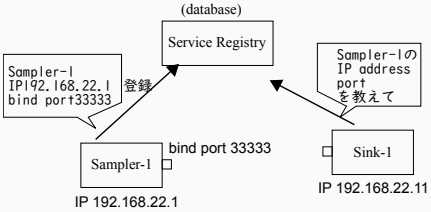
多数のプロセスを動かそうとしたとき

- プロセス同士の接続設定を簡単に行いたい
  - 1 to 1
  - 1 to N
  - N to M
- これまで私が扱ったことのある DAQ(小中規模向け) では
  - HDDAQ : bind ポート番号を決めておいて connect するように記述
  - DAQ-MW : ポート番号は知らなくていいが InPort-OutPort の接続は一つずつ記述する
- ZeroMQ でも socket のアドレス (IP アドレス + ポート番号, UDS name) は設定しないといけない
- 10 個ぐらいならまあ我慢できる. 100 個, 1000 個あったら...

→ **service discovery** の導入

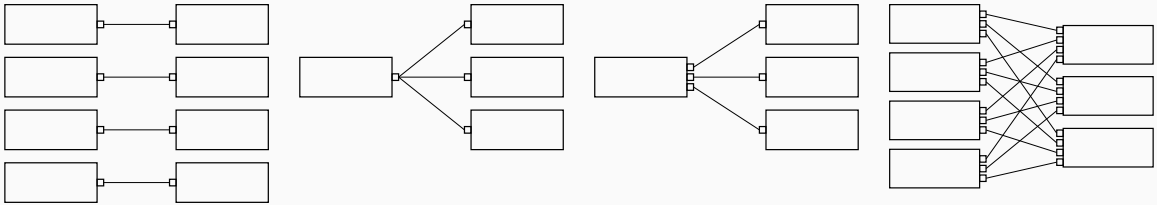
Service discovery

- **service registry**: プロセスのホスト (IP アドレス), ポート番号, 死活状況などの情報を管理するデータベース (Key-Value store)
- 各プロセスは
  - 自分自身の情報を registry に登録し、他のプロセスから見つけられるようにする
  - 自分の通信相手の情報を registry の中から探す
- 有名どころ: ZooKeeper, etcd, Consul
- **Redis で service discovery を実装してみた**
  - bind 側のポート番号は指定がなければランダムに割り当てて bind に成功したら registry にポート番号の情報を登録する



もっと簡単に設定したい

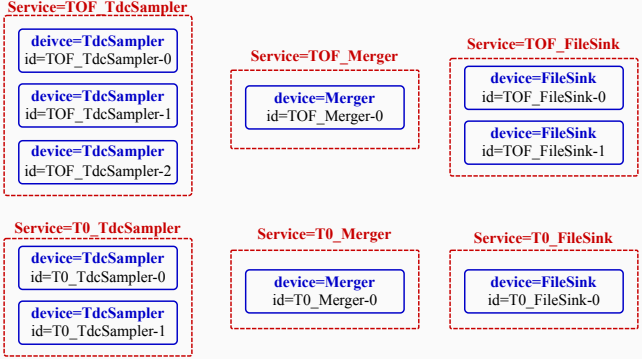
- 個々のプロセス間の接続設定を一つずつ入力していたらラクにはならない
- プロセスのグループ同士の接続設定を与えるとグループに属する個々のプロセス間の設定を自動生成するルールを決める
  - (N 個の) 1 to 1
  - 1 to N, N to 1 (共通アドレス, 個別アドレス=subsocket)
  - N to M



DAQ service plugin (1)：topology configuration

- **Instance**: ユニークな ID を持ったプロセス (FairMQDevice)
- **Service**: 同じ FairMQDevice をまとめた論理グループ
- **Endpoint**: service (instance) の MQ channel (MQ socket)
- **Link**: endpoint 間の通信

service 間の接続情報を Redis に登録すると instance 間の接続情報を自動生成

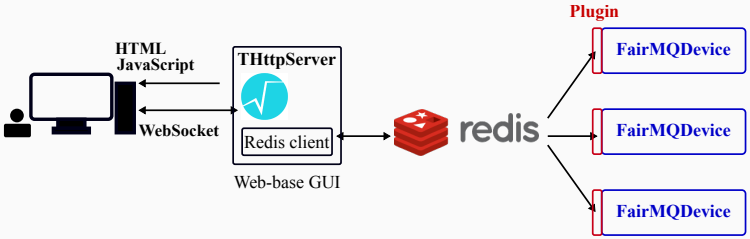


DaqServie plugin (2) : Web-based controller GUI

Redis pub/sub channel を使った状態遷移制御

- Redis のメッセージキューブローカーとしての機能を利用

- シンプルな web-based GUI を開発
  - RUN number の設定, 更新
  - 状態遷移コマンド (Idle → Running, etc.)
- HTML + JavaScript + WebSocket + THttpServer (civetweb). Tested with ROOT 6.18

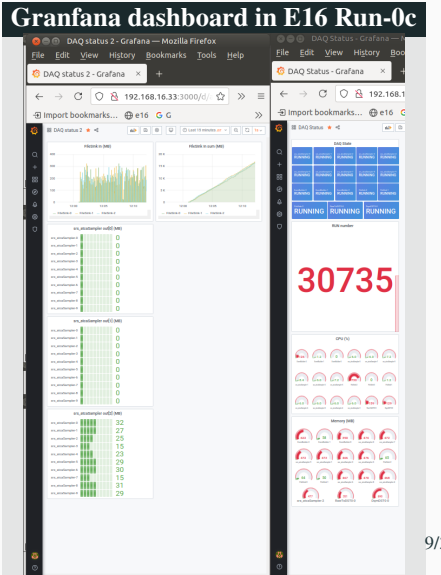


Controller in E16 Run-0c

Metrics plugin

- DAQ state (Idle, DeviceReady, Ready, Running, ...) が変わるときに Redis に保存している state の値を更新する
- Metrics plugin の扱うデータは Redis 拡張機能の TimeSeries data を使用
  - CPU 負荷, RAM 使用率 (reading /proc/stat and /proc/self/stat)
  - メッセージレート (毎秒転送数, 毎秒のメッセージサイズ)
- **Grafana** から Redis にアクセスしてデータを可視化

Granfana dashboard in E16 Run-0c



Redis を使った FairMQ の plugin (まとめ)

DAQ service

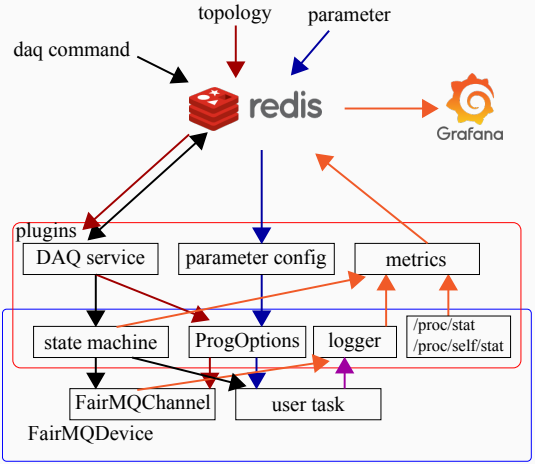
- **service discovery (プロセス間接続設定の補助)**
- ステートマシン制御 ← Redis PUB/SUB
- run 番号設定

Parameter config

- ユーザータスクで使うパラメータの設定補助

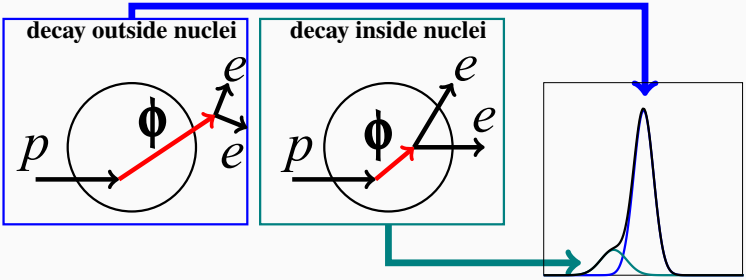
Metrics

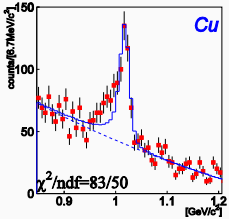
- CPU/RAM の負荷, データ転送量の取得
- Grafana での可視化



ハドロン質量起源の解明

- カイラル対称性の自発的破れ ← 高密度媒質 (=原子核) での回復
- 原子核中のベクターメソンの質量スペクトル変化の系統的測定で調べる
  - p+A(C, Cu, Pb, etc.) 反応. $\rho/\omega/\phi \to e^+e^-$
  - 遅い中間子 & 大きい原子核標的 → 原子核密度での崩壊イベントが起きやすい
- KEK-PS E325: 真空中での崩壊による予想と比べて低質量側に excess
- 質量変化現象の確立, 原子核内のクォーク凝縮, 分散関係 → 高統計・高分解能な実験が必要

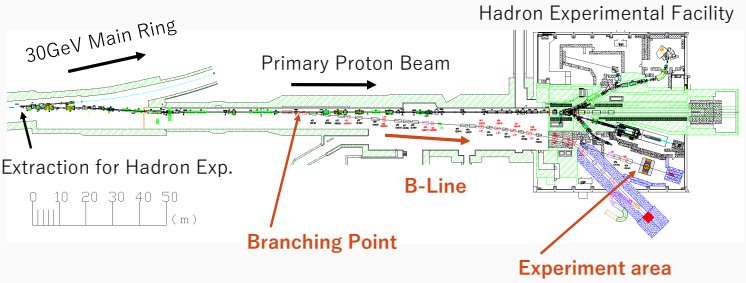


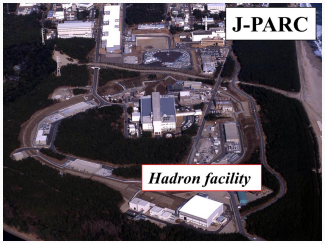


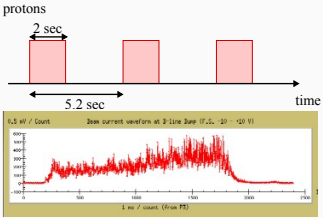
KEK-PS E325
ϕ の質量スペクトル

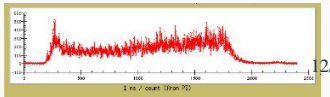
J-PARC ハドロン実験施設 高運動量ビームライン (high-p BL)

- 遅い取り出し 1 spill = 2.0 sec, 繰り返し 5.2 sec (2021)
  - 将来的には繰り返し周期が短くなる可能性
- 30 GeV 1 次陽子ビームの一部 (10^{10}/spill) を分岐して実験標的まで輸送









J-PARC E16 実験の検出器

- 反応レート 10^7 Hz での e^+e^- 精密測定
- 双極磁場中に 26 個のモジュールをパレル状に配置

飛跡検出器

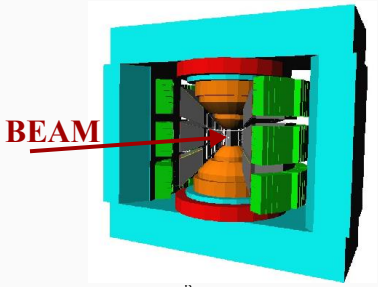
- SSD (Silicon Strip detector) ~53,000 ch (2,048 ch/module)
- GTR (GEM Tracker) ~58,000 ch (2,232 ch/module)

電子同定用検出器

- HBD (Hadron blind detector) ~36,000 ch (1,400 ch/module)
- LG (leadglass EMCAL) ~1,100 ch (38–42 ch/module)

staging strategy

- **Run0 (2020, 2021) :**
  6 (SSD) + 6–8 (GTR) + 4–6 (HBD) + 6 (LG) → コミッショニング (今ココ)
- Run1 (2022–) : 8 + 8 + 8 + 8 → 物理測定, C/Cu 標的
- Run2 (???) : 26 + 26 + 26 + 26 → 物理測定, C/Cu/Pb 標的



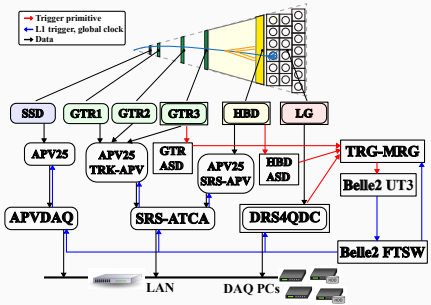
Run0 のトリガー・読み出し回路

トリガー回路

- discri-out → TDC を高速シリアル転送で集約 → トリガー判定
  - e-track candidate = GTR-ASD × HBD-ASD × LG
  - vector meson = 60 deg 以上開いた e-track が 2 個
  - L1 request ~1 kHz
- B2TT による信号分配 (250 Mbps, 8b10b シリアルデータ)
  - グローバルクロック: 125 MHz
  - トリガー 4 bit: L1, spill-start/-end, monitor, ...
  - トリガータグ 96 bit: timestamp (48b), spill count (16b), event count (32b)
  - BUSY 収集

読み出し回路: アナログメモリによる波形サンプリング

- SSD (APVDAQ, VME) : APV25 40 Msps, 8(9) samples
- GTR, HBD (SRS-ATCA) : APV25 41.66 Msps, 24 samples
- LG (DRS4QDC): DRS4 960 Msps, 200 samples

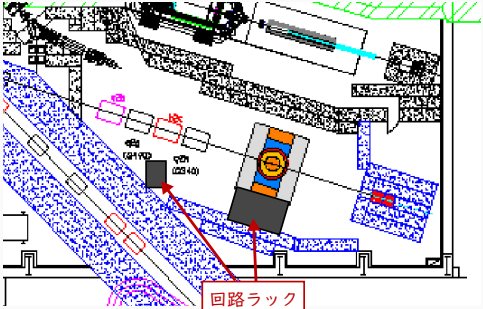


検出器モジュール 1 台あたりの trigger primitive 信号の数

- GTR-ASD : 24 ch
- HBD-ASD : 36 ch
- LG-discri : 38–42 ch

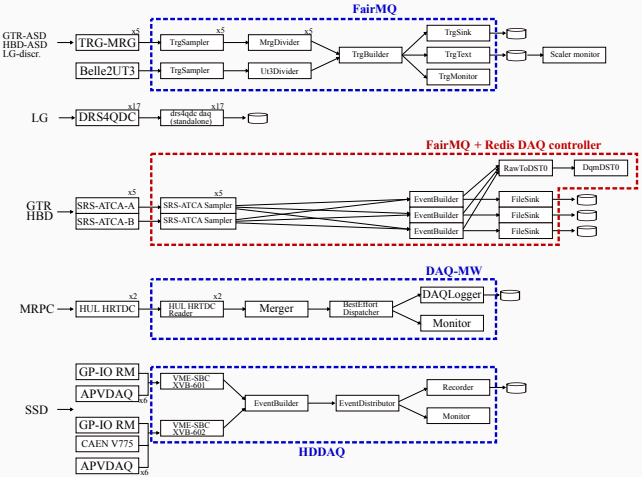
実験エリアの様子

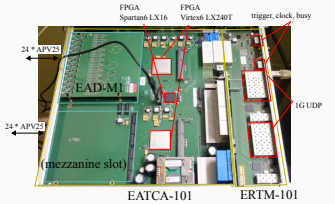
- APV25, GTR-ASD, HBD-ASD は検出器近傍, APV25 の repeater ボードは上流側に設置
- 上記以外 (デジタイザー, トリガーロジック, slow control, HV 電源, LV 電源, etc.) は磁石ヨークの脇の回路ラックに設置
  - Xilinx 6/7 series の FPGA には放射線対策として SEM コアを実装. (config RAM 1 bit のエラーまでなら訂正)
  - ビームダンプに近いラックに設置した Spartan-6 はエラーが発生しやすかった
- 検出器 (フロントエンド回路)–回路ラック : 15–20 m
- 回路ラック–DAQ 計算機 (ハドロン南実験棟) : ~150 m の光ファイバーで接続
- ほとんどの回路の AC100V 電源は遠隔で On/Off できるようにした

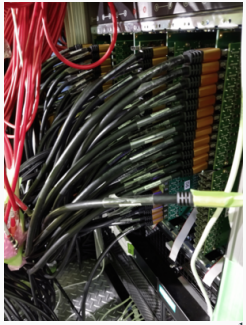


DAQ in E16 Run-0c

- Trigger, DRS4QDC, SRS-ATCA → 1 PC (Xeon E5-2630v4 10 cores × 2)







Streaming DAQ soft. for COMET extinction meas. was used for time structure of trigger signals. (not shown)

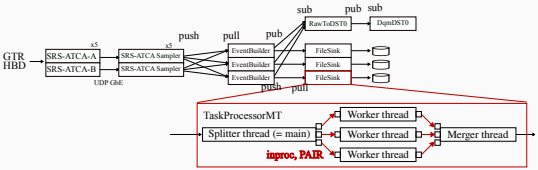
DAQ software for SRS-ATCA using FairMQ + Redis

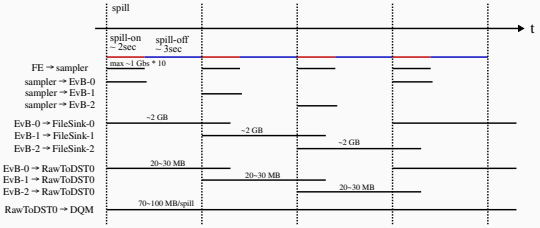
Data path

- **push-pull** connection
- 10 samplers, 3 event builers, 3 file sinks : up to ~1 Gbps/sampler (GbE from SRS-ATCA)
- **Spill-ID based round robin**
- Multithreaded in FileSink to compress data with ZSTD

Monitor path

- **pub-sub** connection
- 1 data converter (raw data to DST0 hit/event object data), 1 data quality monitor





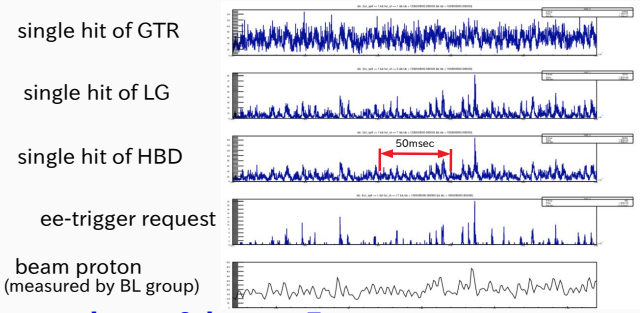
Problem

- Low DAQ live time : 15% (Run0c) for L1 request rate of 1 kcps
- Caused by beam structures?
  - 5 ms ← MR power supply?
  - 5.2 μs (= 191 kHz J-PARC MR freq.) ← dispersion @ Lambertson magnet?

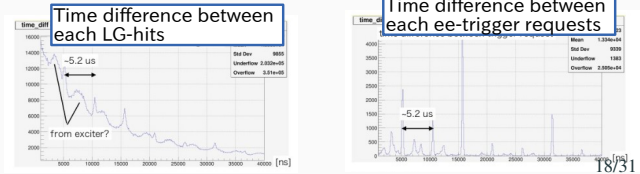
Countermeasure

- Firmware update to reduce dead time
- Bug hunting of trigger logic
- Hardware upgrade
- Long term plan : gradually move to streaming DAQ

trigger & beam: 5 ms structure



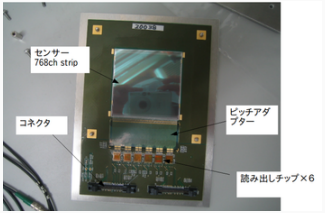
trigger & beam: 5 us structure



Upgrade of silicon strip detector and readout electronics

Problem

- SSD and APVDAQ in Run0 was temporarily borrowed from J-PARC K1.8 group



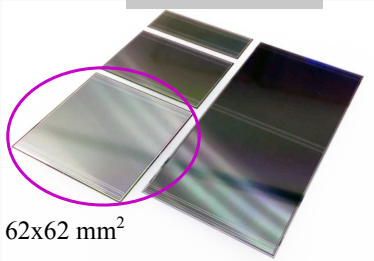
SSD in Run0

APVDAQ

Run1: Use CBM(GSI/FAIR)'s SSD

- double-sided SSD : 62×62 mm^2
- 1024 ch×2
- stereo angle : 7.5 deg.
- strip pitch : 58 μm

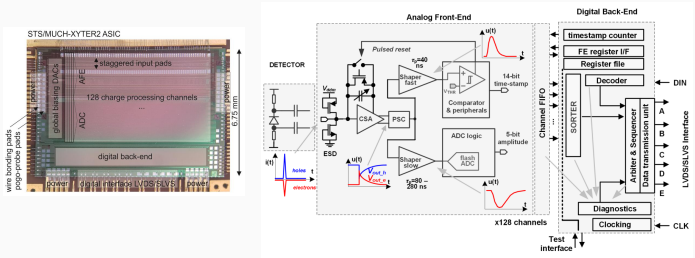
CBM STS sensor



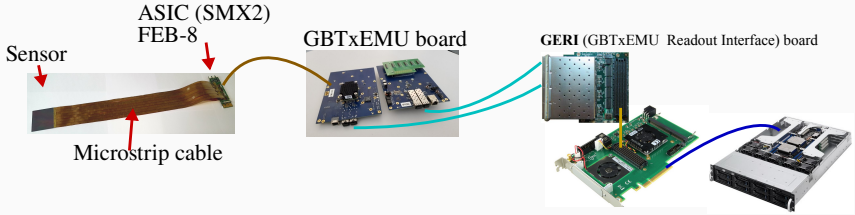
62x62 mm^2

SSD frontend ASIC : SMX2 (STS-MUCH-XYTER2)

- UMC 180 nm CMOS
- input 128 ch (+ 2 ch test)
- CSA gain switchable (STS=0–16 fC, MUCH=0–100 fC), polarity select(positive/negative)
  - cf. CBM-STS(Silicon Tracking System) = 1.8M ch, CBM-MUCH(Muon Chamber, GEM) = 249k ch
- **Self-triggered readout of hit data**
  - **14 bit TDC** (fast shaper + leading-edge discri.), LSB = 3.125 ns @ 160 MHz clock
  - **5 bit peak ADC** (slow shaper)
- LVDS/SVLS link (AC coupled 8b10b)
  - down-link (input, multi-drop) : 160 MHz, control
  - up-link (output) : 1–5 links, 320 Mbps/link, 9.4 M frames/sec/link
- Power consumption: <10 mW/ch (~1 W/chip)



New SSD readout system of J-PARC E16



Closer to BM@N(JINR)'s system rather than CBM's GBTx-based CROB + BNL712

- low-mass and low capacitance microcable
- FEB-8 (Frontend board) : SMX2 8 chip, 2 up-links/chip
- GBTxEMU : FMC carrier + SoM (TE0712)
  - Self-triggered readout from SMX2 to GBTxEMU
  - Hit selection by ADC value and timestamp
  - GBT 4–5 Gbps × 1–2, IPbus
- GERI board : COTS PCIe readout board

To do

- Cooling system
- Cable adatper for long transmission from FEB-8 to GBTxEMU
- global clock + trigger I/F (B2TT, CDCM)
- Firmware customization
  - DAQ with trigger (Run1)
  - Streaming readout (Run2?)

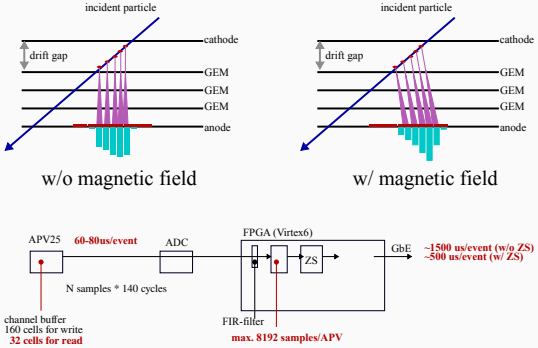
GEM readout problems: DAQ bottlenecks

Data (analog) transfer: APV25 → FPGA (SRS-ATCA)

- **60–80** μs for 24 samples readout
- A drift time measurement is important to achieve good position resolution. (μ-TPC reconstruction method)
- slower drift velocity → better position resolution but longer drift time
- A time window of at least ~430 nsec (18 samples @41.66 Msps) is needed to keep good detection efficiency.

Data transfer: FPGA (SRS-ATCA) → PC

- **0–1.5 msec**: depends on number of hits, buffer occupancy in FPGA



GEM readout in Run1: Firmware modification of SRS-ATCA

APV25 → FPGA

- Optimize number of samples per event (and sampling speed)
- <15 samples → The read buffer in APV25 can hold multiple events.
- **0–70** μs

FPGA → PC

- Attach DDR3-SODIMM modules to SRS-ATCA
- 1 GbE (UDP) → 10 GbE (XAUI, UDP)
- **No deadtime**

DAQ live time: 10–20% → 50–70% (expected)

Streaming readout of GEM: Run2(?)

SMX2 for GEM

- Pros: R&D work is shared with SSD readout upgrade.
- Cons: Cooling system, cables → more installation space, background source

MPGD ASICs

- Waveform: SAMPA (ALICE), WASA (CEPC, in progress)
- HR-TDC(50–100 ps TAC) + ADC (or ToT): VMM3 (ATLAS), TIGER (BES III)

How about the development of a new ASIC (low power, high throughput, high performance) in Japan?

Upgrade of EMCAL readout (1)

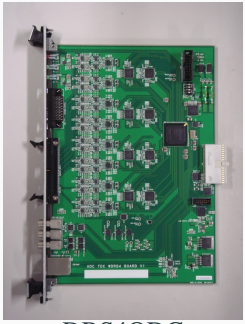
Problems with DRS4QDC

Production

- High failure rate of board assembly
- EOL of Xilinx Platform Flash PROM. We can still purchase but ...

Performance/specification

- 2 μs L1 trigger latency due to analog memory depth
- Dead time due to data transfer from DRS4 to FPGA = 6–10 μs (RoI 100-200 samples)
- Max throughput = 100 Mbps.
- Complicated data correction
  - Cell-by-cell offset correction
  - Cascading capacitor arrays
  - Removal of symmetric spike noises (Cells around the end of RoI are biased by read-address-line)
  - Offset of cells in RoI are shifted. Recovery time ~100 msec.



DRS4QDC

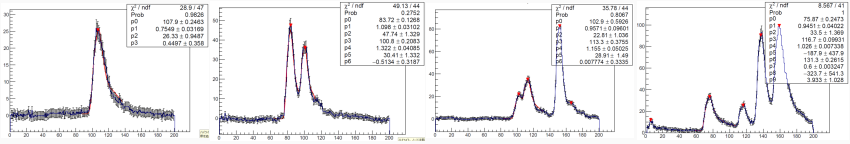
- Run1:
  Continue to use DRS4QDC
- Run2:
  Upgrade if possible

Upgrade of EMCAL readout (2)

Still under discussion within E16 group

Plan 1: fast ADC (→next page)

- Continue to use the existing LG. Waveform readout is essential.
- 500–1000 MSPS, 10 bit ADC



Examples of LG signal (DRS4QDC. RoI~200 nsec)

Plan 2: Fine segmented EMCAL (not included in this talk)

- Reduce the hit rate per channel
- ~150×150 mm^2 → e.g. ~20×20 m^2. increase of number of channels : ×50
- LG → XXX
- Collaborate with J-PARC HI (Heavby Ion)?

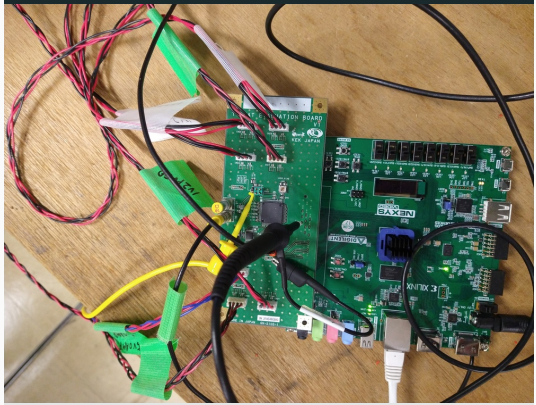
A candidate for fast ADC: TIAMAT (OpenIt)

TIAMAT present status (Miyahara-san, Hamada-san)

- TSMC 65 nm CMOS
- Time-interleaved sampling, input 1 ch
- Sampling freq: 1 Gsps
  (800 Msps confirmed)
- Resolution: 10 bit
- Data output I/F: 10 LVDS (data 8 pairs, clock 2 pairs)
- Control I/F: SPI
- Low Power consumption: ~300 mW/chip

- Option of 500 Msps×2 ch?
- Advantages compared to COTS ADC(ADI, TI)?
- RFSoC is too expensive...

TIAMAT test boad (FMC LPC) + Nexys Video



A test with LG is planed.

まとめ

- J-PARC E16 実験の計測システム : 高計数率環境での e^+e^- 精密測定
- Run0 (2020/Jun, 2021/Feb, 2021/Jun) : コミッショニング
  - APV25, DRS4QDC による波形取得
  - DAQ ソフトウェア: FairMQ + Redis (service discovery, control, etc.) を部分的に使用

課題

- ビームの時間構造で DAQ live time が低下している可能性が高い

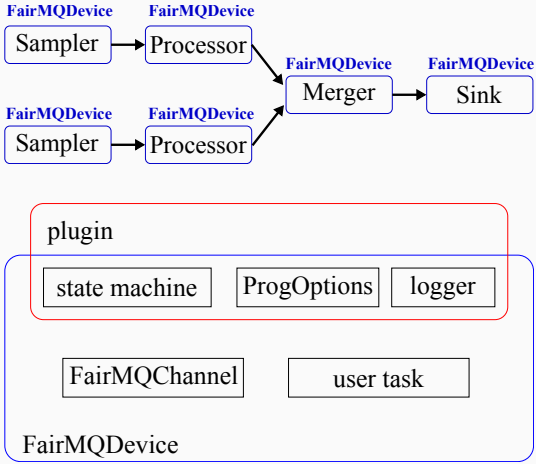
今後

- 既存回路の fimrware 改良による deadtime 削減
- 将来的には streaming DAQ へ移行したい (個人の感想です)

Backup

ソフトウェア構成 (プロセス内構成)

- **FairMQDevice**
  - ワークフローを構成する単位. (プロセス)
  - 仮想関数を override してユーザーのタスクを実行する
- **FairMQChannel (FairMQSocket)**
  - Device 間のデータ通信を行う.
    具体的には ZeroMQ socket など
- **Plugin**
  - state machine の制御, 監視
  - **ProgOptions** パラメータの設定, 監視
  - 外部ツール (UI, database) との連携
- **FairLogger**
  - プロセスの実行ログを管理
  - 重要度, 詳細, 出力先 (console, file, 外部プロセス)



user task と plugin の間でのデータの受け渡しは, 基本的には ProgOptions または logger で行われる.

FairMQDevice: State machine

- 水色の各 state で実行される仮想関数があるので, それらをカスタマイズする.