HL-LHC ATLASミューオントリガーシミュレータの開発及び、それを用いた性能評価と実機の検証

浅井研究室 修士課程2年 山下恵理香

LHC-ATLAS実験

LHC加速器(Large Hadron Collider)

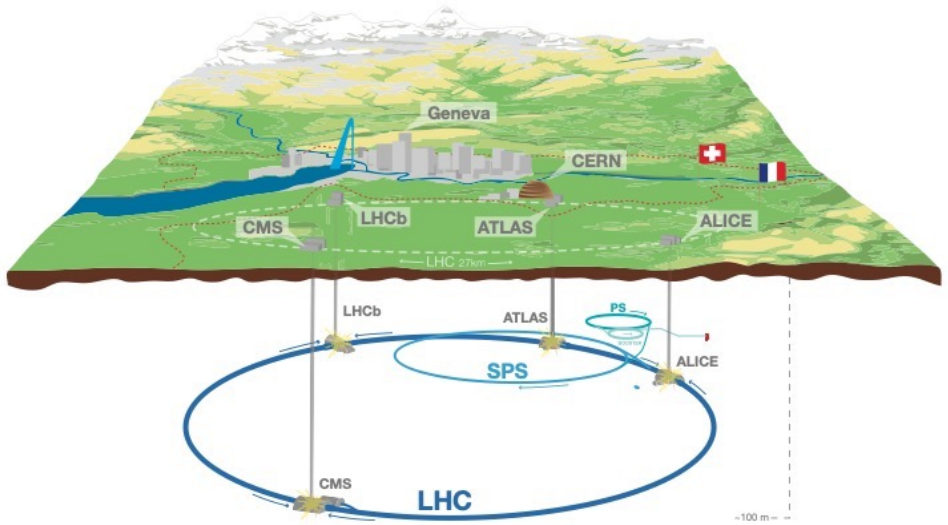
- 世界最高エネルギーの13.6TeV(Run3)で陽子・陽子衝突を起こし、新物理を探すエネルギーフロンティア実験
- 40 MHzで陽子を交差させる

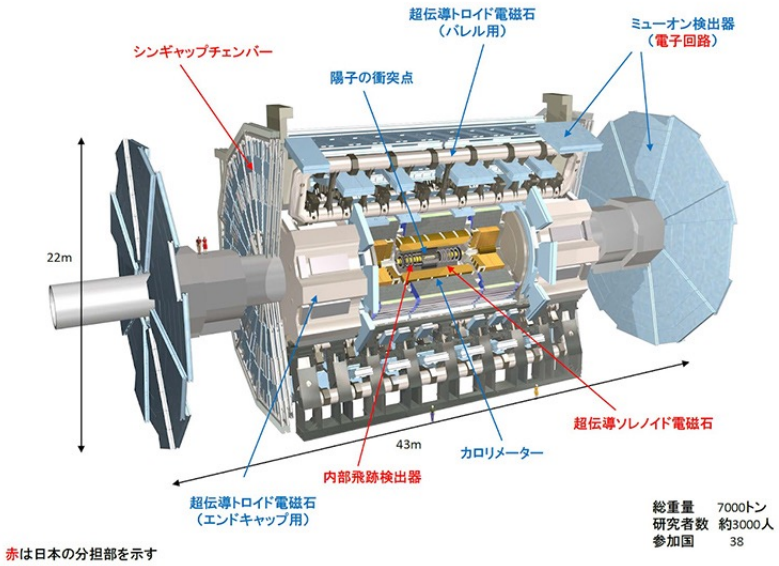
ATLAS検出器

- LHCの陽子・陽子衝突で生成された粒子を観測するための大型汎用検出器
- 複数の種類の検出器で構成される

観測したい物理

- 標準模型の粒子の精密測定
- 超対称性理論などで予言される、暗黒物質などの新粒子の探索





ATLAS検出機の全体図 (https://atlas.kek.jp/main/research_summary/index.html)

LHC-ATLAS実験

LHC加速器(Large Hadron Collider)

- 世界最高エネルギーの13.6TeV(Run3)で陽子・陽子衝突を起こし、新物理を探すエネルギーフロンティア実験
- 40 MHzで陽子を交差させる

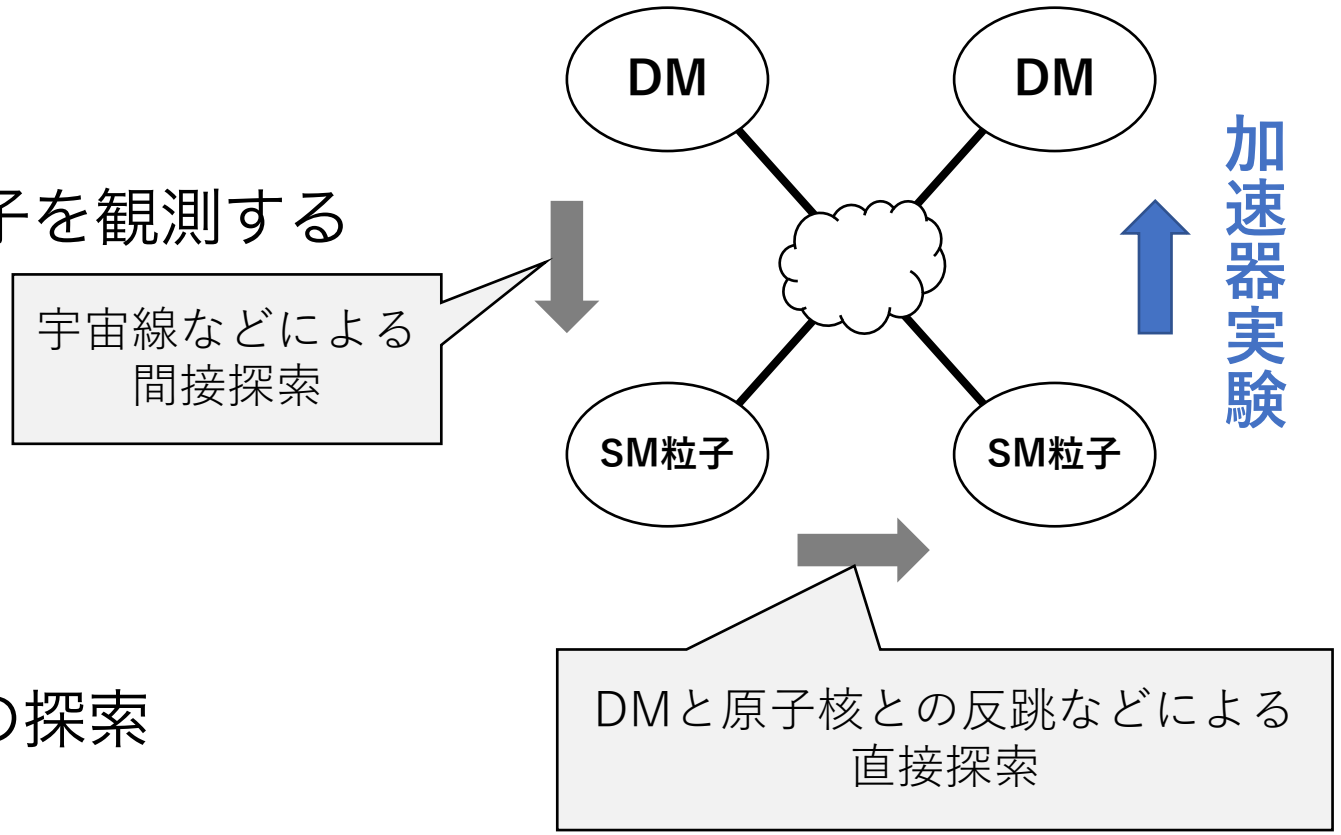
ATLAS検出器

- LHCの陽子・陽子衝突で生成された粒子を観測するための大型汎用検出器
- 複数の種類の検出器で構成される

観測したい物理

- 標準模型の粒子の精密測定
- 超対称性理論などで予言される新粒子の探索
  - 暗黒物質等も含む

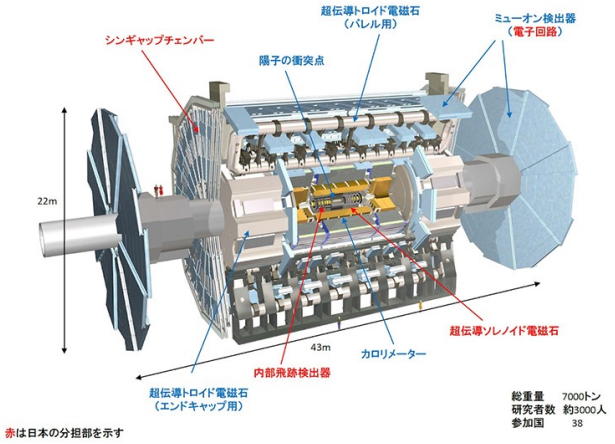
例：暗黒物質探索



LHC-ATLAS実験のトリガーシステム

トリガーシステム

- 40 MHzの陽子交差の内、ほとんどは物理的な興味の薄い陽子の非弾性散乱
  →トリガーによってオンラインで記録する事象を選別する
- ATLASのトリガーは2段階で行われるが、
  本研究の対象は初段トリガーの一部のエンドキャップミューオントリガー
  - 以下は2029年以降の値
  - 10 μsレイテンシーでミューオンの飛跡を再構成し、事象選別に活用する

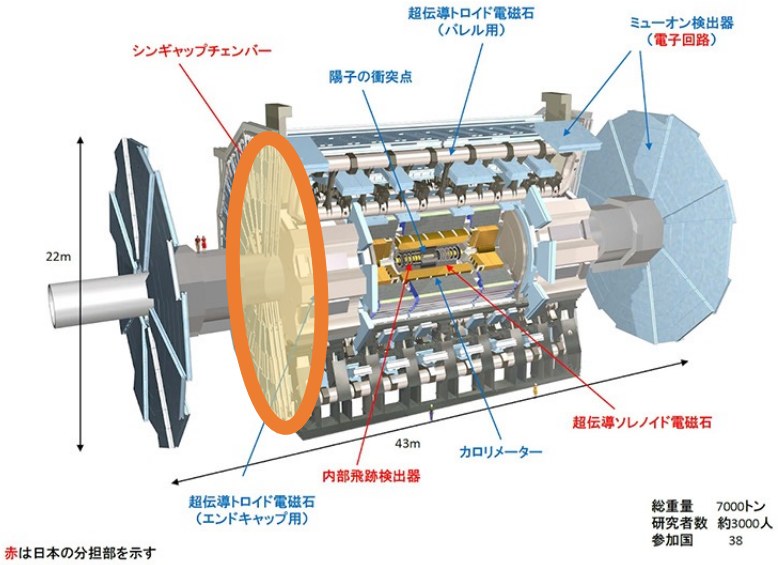


40 MHz → **初段トリガー** → 1000 kHz → **後段トリガー** → 10 kHz → **記録**

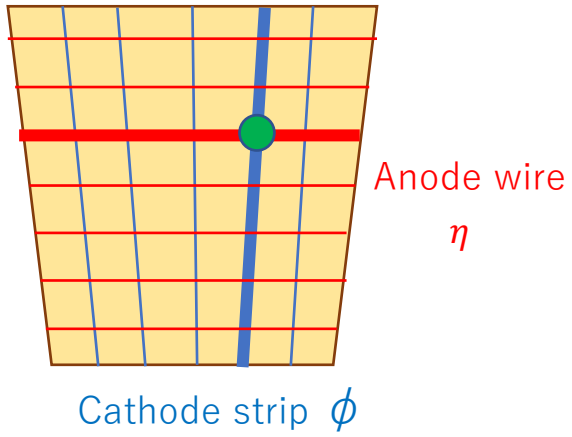
TGC検出器とトリガー

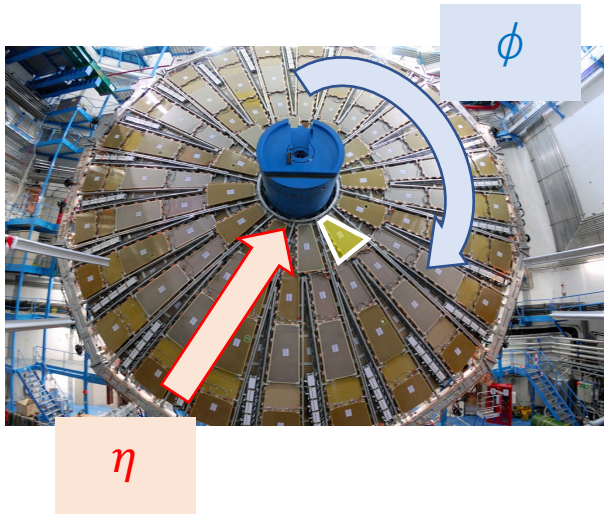
TGC(Thin Gap Chamber)とエンドキャップミューオントリガー

- 高い横運動量p_Tを持つミューオンを含む事象を選別するためのトリガーとして使用
- アノードワイヤーとカソードストリップによる二次元読み出しを行うMWPC
- 7層・3ステーションで構成され、コインシデンス回路で高速飛跡再構成（直線飛跡再構成）を行う
  - Wire 7層+Strip 6層
- トロイド磁場による曲率と直線飛跡との差分$(d\eta, d\phi)$の相関関係を利用し、横運動量p_Tを計算する



ATLAS検出機の全体図

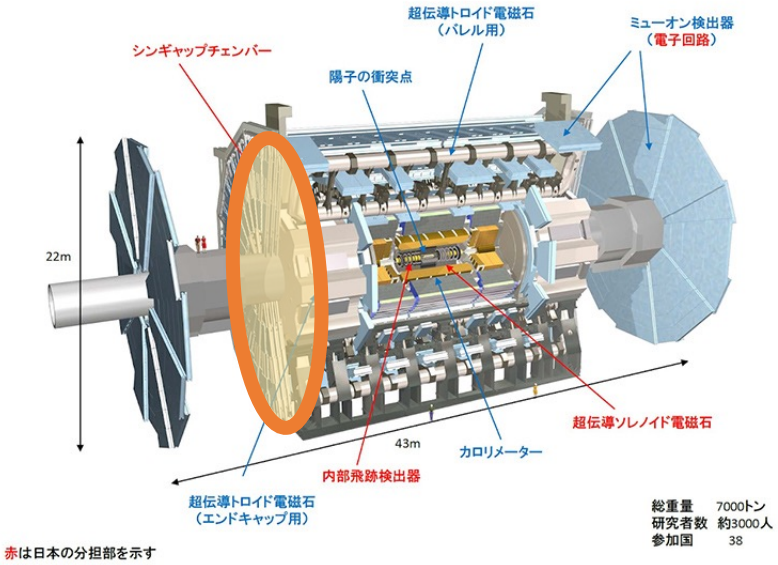




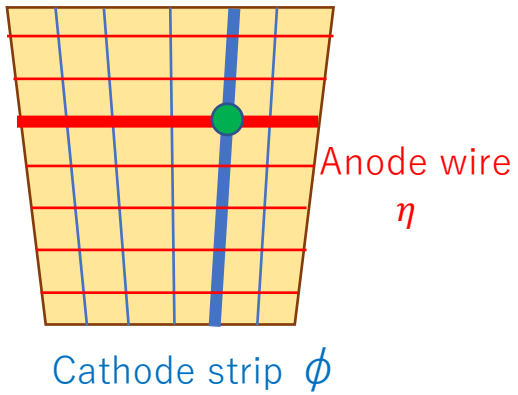
TGC検出器とトリガー

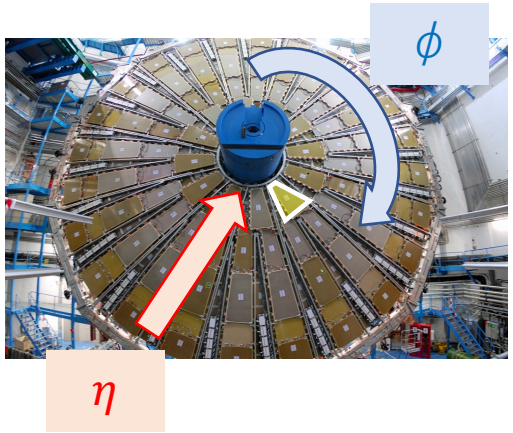
TGC(Thin Gap Chamber)とエンドキャップミューオントリガー

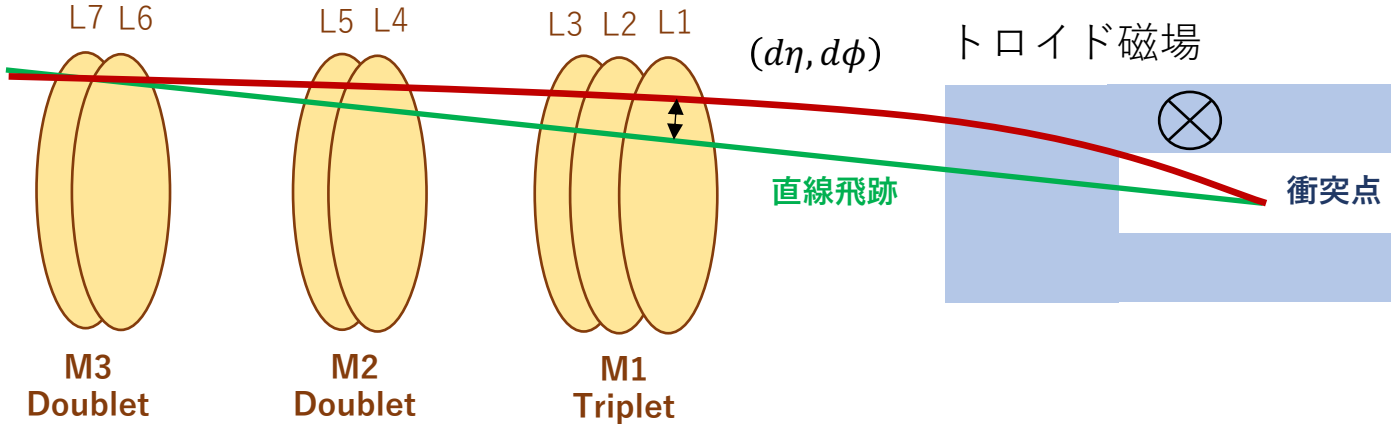
- 高い横運動量p_Tを持つミューオンを含む事象を選別するためのトリガーとして使用
- アノードワイヤーとカソードストリップによる二次元読み出しを行うMWPC
- 7層・3ステーションで構成され、コインシデンス回路で高速飛跡再構成（直線飛跡再構成）を行う
  - Wire 7層+Strip 6層
- トロイド磁場による曲率と直線飛跡との差分$(d\eta, d\phi)$の相関関係を利用し、横運動量p_Tを計算し、トリガー判定に用いる



ATLAS検出機の全体図

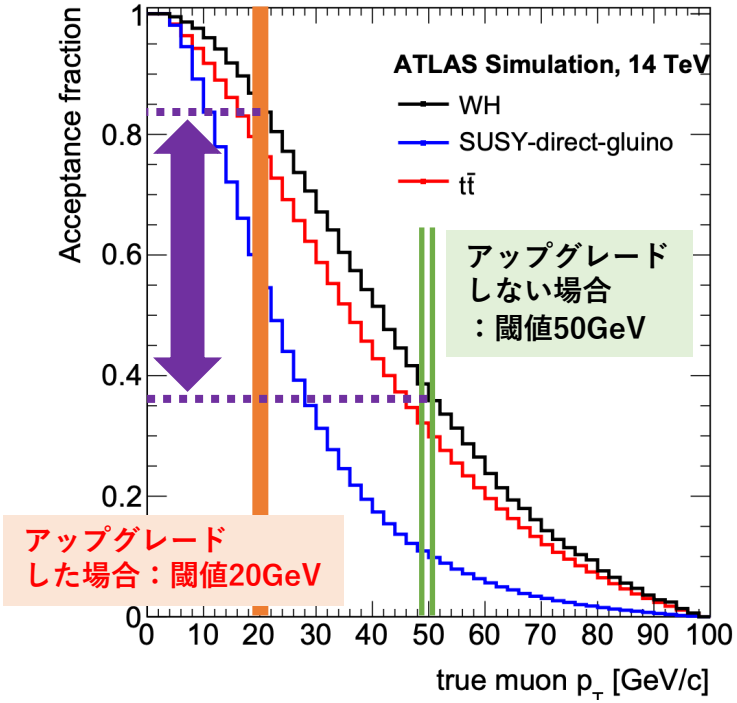






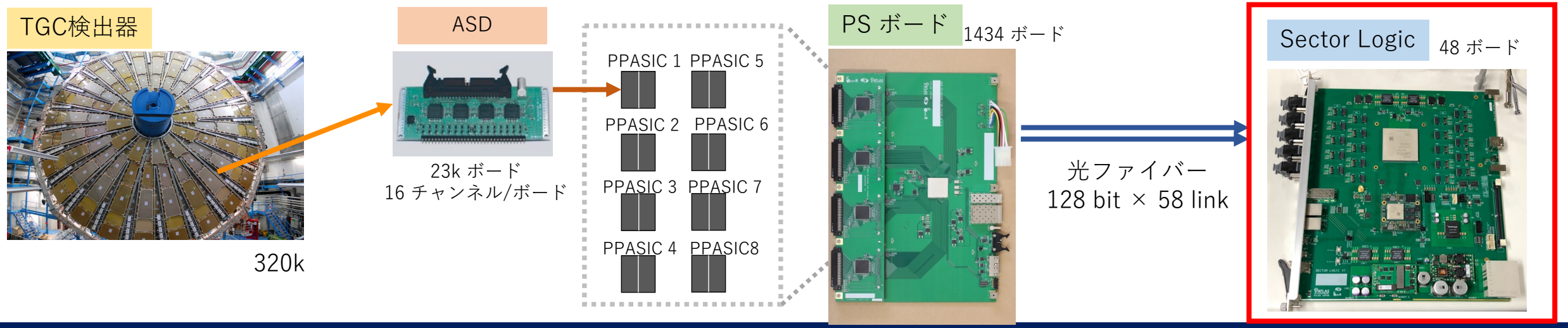
高輝度 LHC-ATLAS 実験に向けたPhase2 Upgrade

- 現在は物理実験期間中(Run3)
- 2029年から始まる高輝度LHC実験（Run4）に向けたアップグレードが進行中
  - 瞬間ルミノシティが$5 \sim 7\times10^{34} cm^{-2}s^{-1}$ になる
    - ATLAS検出器設計当初の5~7倍
  - 高輝度環境に対応するため、エンドキャップミューオントリガー系のエレクトロニクスが刷新される→Phase2 Upgrade
- Phase2アップグレード後のエンドキャップミューオントリガー系
  - 実験室ではデータの選別をせず、全データをSector Logic(別室にある後段のエレクトロニクス)へ送信する
  - Run3よりもトリガー計算の時間(レイテンシー)を長く取り、複雑な計算が可能になった

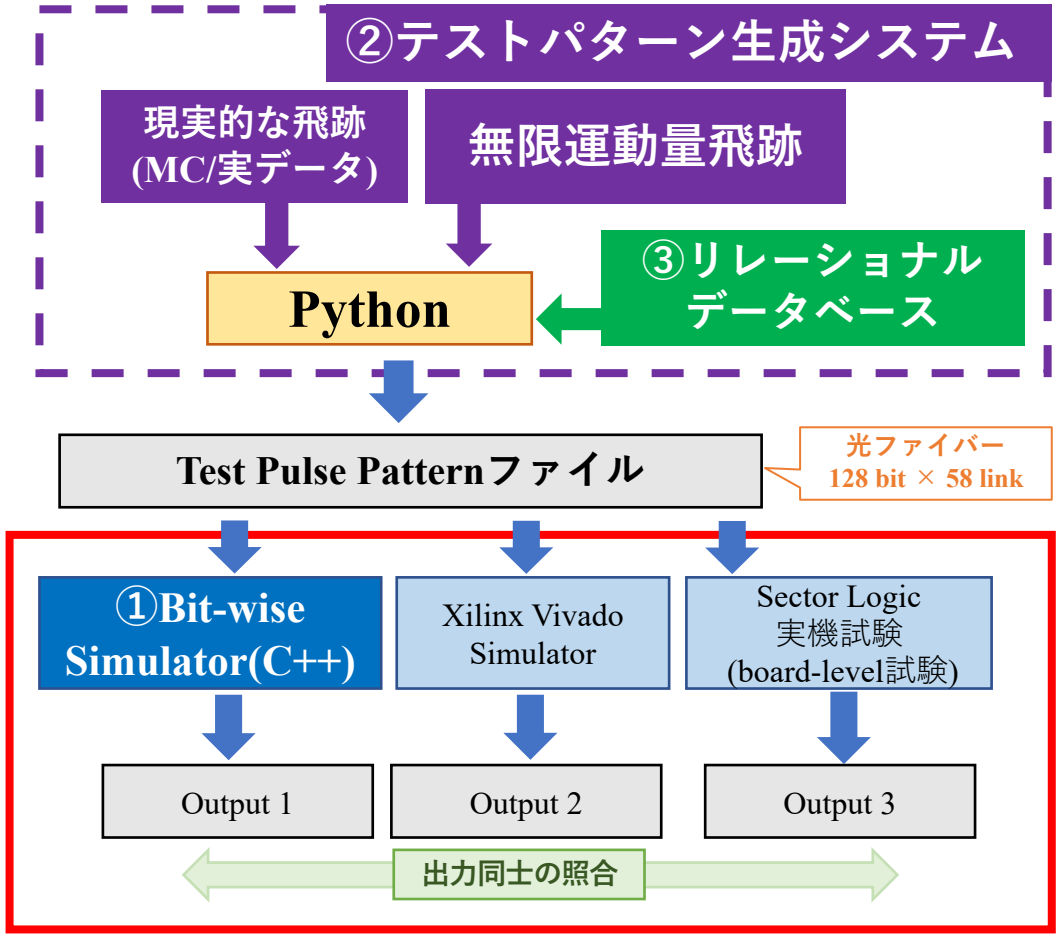


高輝度 LHC-ATLAS 実験に向けたPhase2 Upgrade

- TGC検出器で観測したヒット情報は、多段のコンポーネントを経てトリガー判定される
  - ASD　　信号のデジタル化
  - PSボード　　LHCクロックと同期
  - Sector Logic(SL)　　7層のコインシデンスによりp_{T}を計算
    - トリガー系では、データの選別をせずに全情報をSLへ送信する
- PSボード以降の全システムがエレクトロニクス・ファイバー接続を含めて刷新
  - SLのトリガー系は開発中であり、現在も統合試験を行っている
  - 本研究はSL開発・運用のためのトリガー系検証機構の全体設計+開発+利用を行った



SL検証機構の全体設計



開発段階(現在)～SL運用時までを見据えて、Phase2アップグレードのインフラストラクチャとしてSL検証機構の全体設計と構成要素の開発を行なった

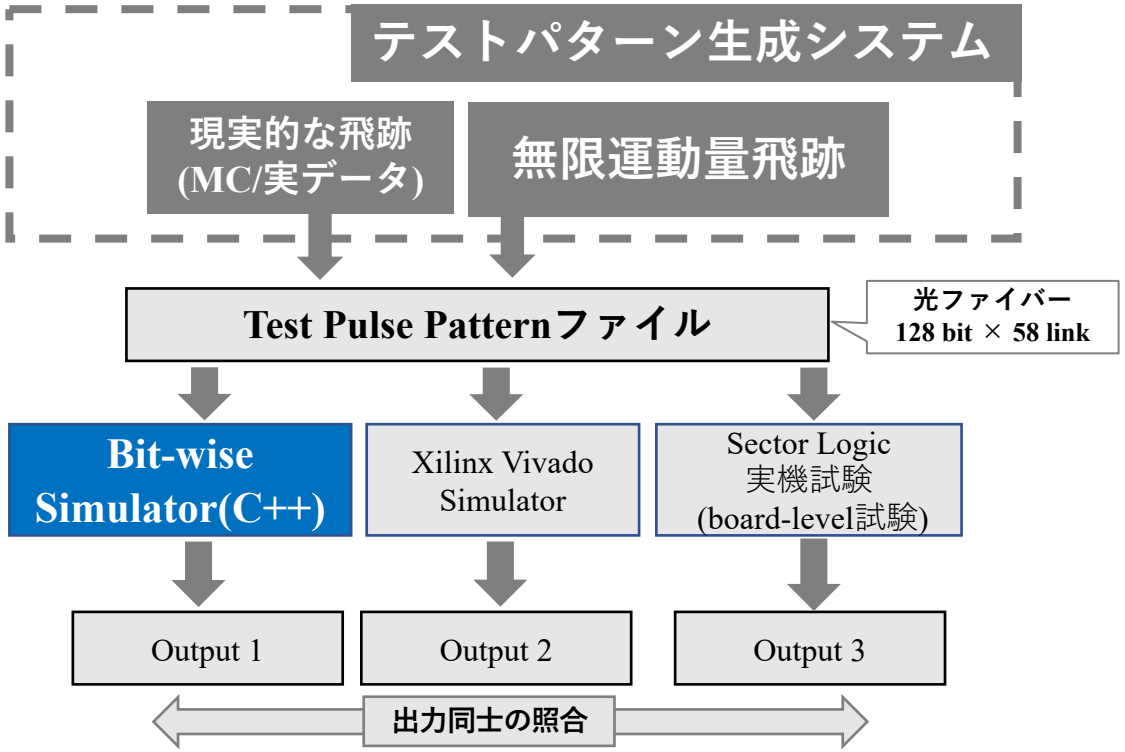
- ソフトウェアとハードウェアの相互検証による堅固な検証機構を設計した
  - 共通のテストパルスパターンファイルを入力として、ソフトウェアシミュレーションも、FPGA論理回路のSimulator(Vivado)も、Sector Logicボードの実機試験もできるようにし、相互検証が自由自在に行える全体設計を作った

検証機構の要素として、以下の開発を行った

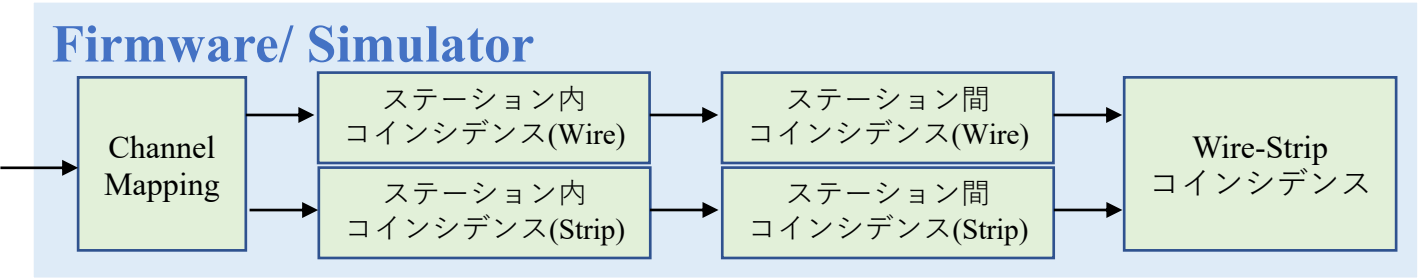
1. Bit-wise simulator
   - SLトリガー系のファームウェアと完全に同一のロジック・入出力
   - C++で記述され、イベント毎の計算がファームウェアシミュレータよりも早い
2. テストパターン生成システム
   - 無限運動量飛跡やMCデータを利用した現実的な飛跡など、さまざまなテストパターンを作成可能
3. リレーショナル・データベース
   - コンポーネント間のケーブリング情報などを一元管理する

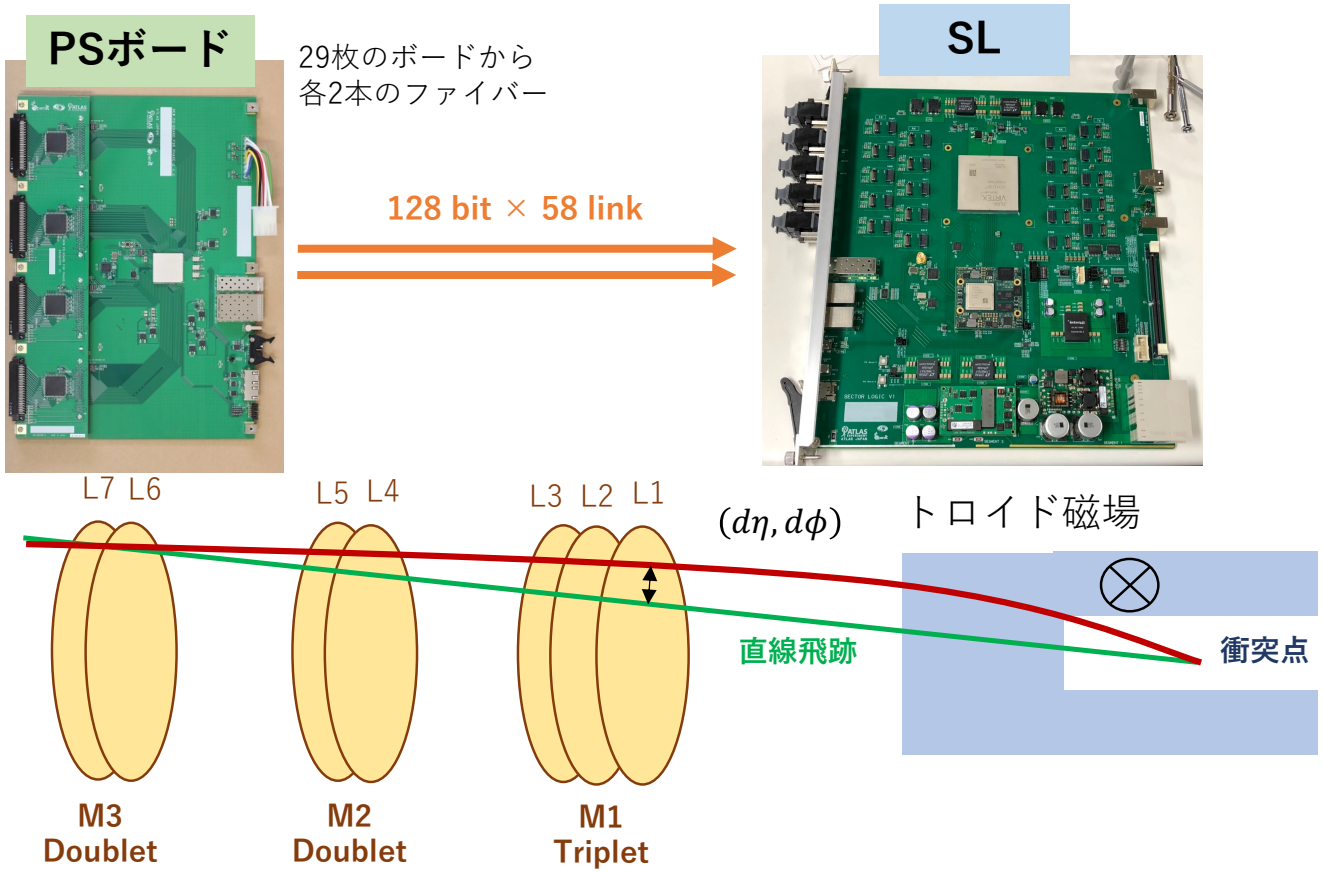
シミュレーションから実機試験までを系統的に動作検証を行うことができる研究基盤を構築し、実際に運用している

セクターロジックトリガー系のビットワイズシミュレータの開発



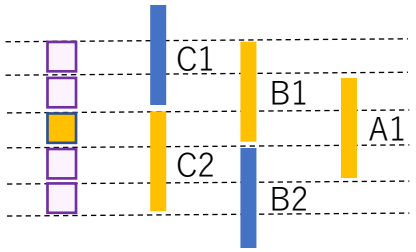
SLのトリガー系の7層コインシデンスの計算



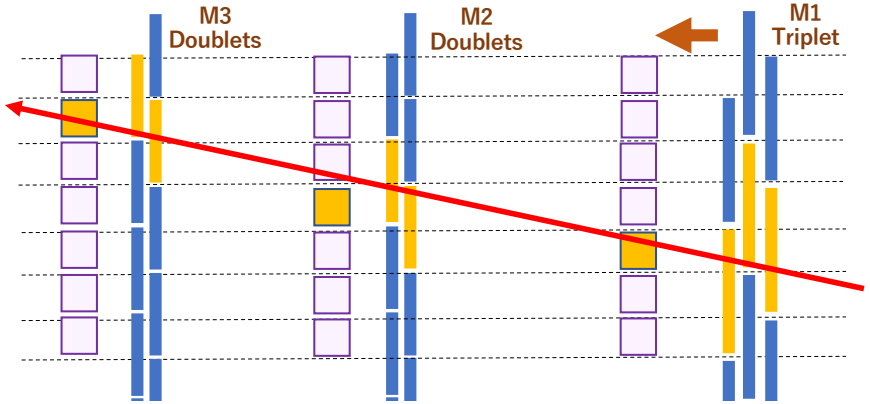


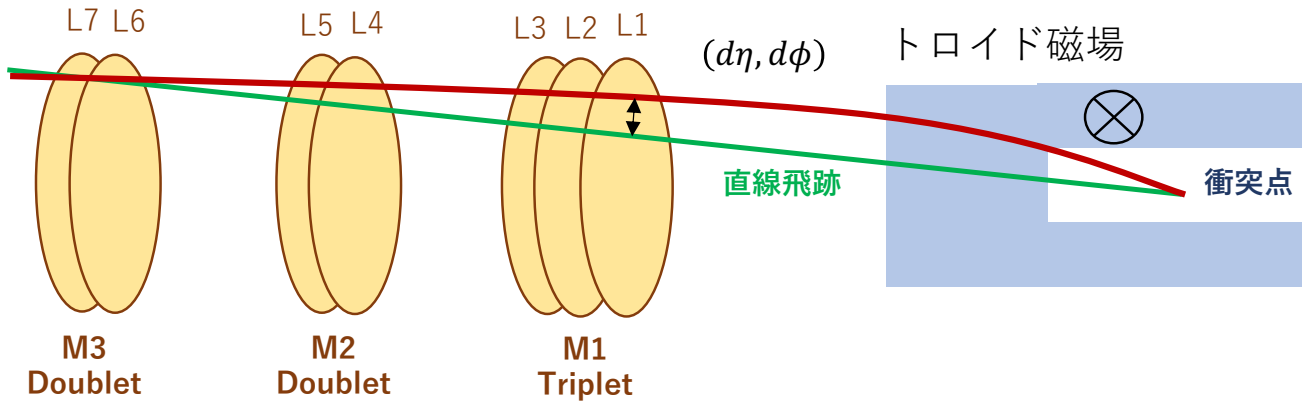
- SLトリガー系では多段の計算によってp_Tを得る
- Channel Mapping
  - トリガーロジックで扱いやすいように下準備をする
    - SLがシリアルリンクで受け取ったビットマップ (128 bit × 58 link) を コインシデンスのためのロジカルなチャンネル（各レイヤーで1次元配列）に変換する
  - この部分のみ、ファームウェアの開発も行った
- ステーション内コインシデンス
  - 各ステーションの代表点を得る
    - ロジカルなチャンネルに対して、AND、OR、NOTで記述されるコインシデンス論理をかける
- ステーション間コインシデンス
  - M1・M2・M3の代表点の組み合わせから、LUT(Look up table)によって無限運動量飛跡との差分$(\Delta\eta, \Delta\phi)$を得る
- Wire-Strip コインシデンス
  - $(\Delta\eta, \Delta\phi)$からLUTによってp_Tを得る

SLのトリガー系の7層コインシデンスの計算



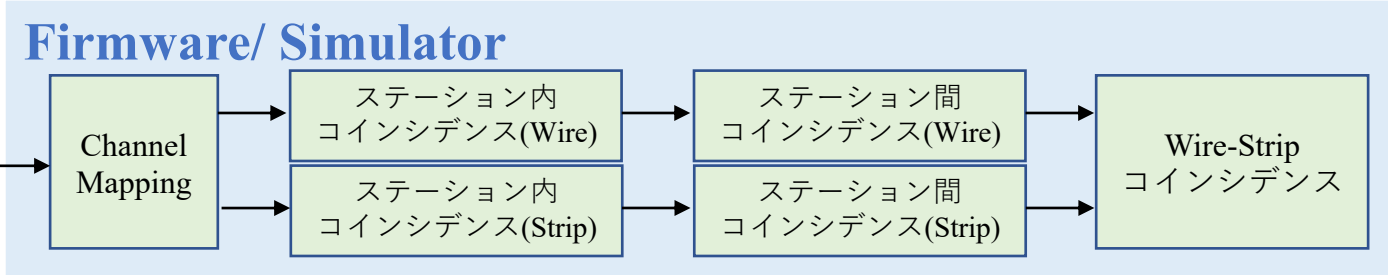
$$3/3 = A1 \,\&\, B1 \,\&\, C2$$
$$2/3 = A1 \,\&\, B1 \,\&\, \overline{C1} \,\&\, \overline{C2} + A1 \,\&\, \overline{B1} \,\&\, \overline{B2} \,\&\, C2 + \overline{A1} \,\&\, B1 \,\&\, C2$$
$$1/3 = A1 \,\&\, \overline{B1} \,\&\, \overline{B2} \,\&\, \overline{C1} \,\&\, \overline{C2}$$

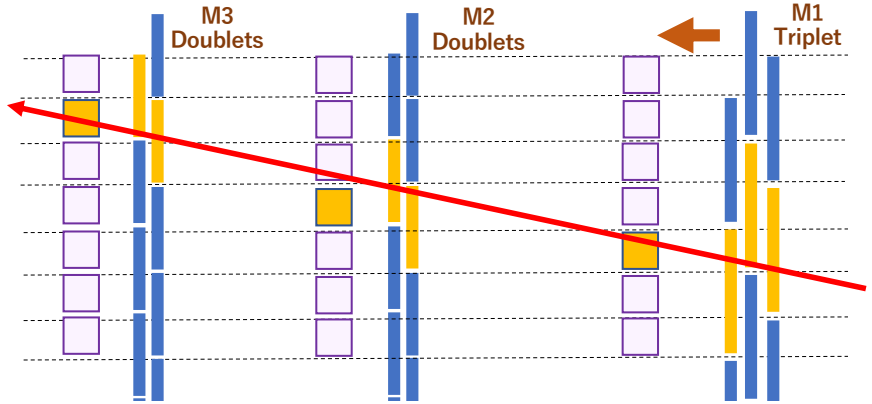


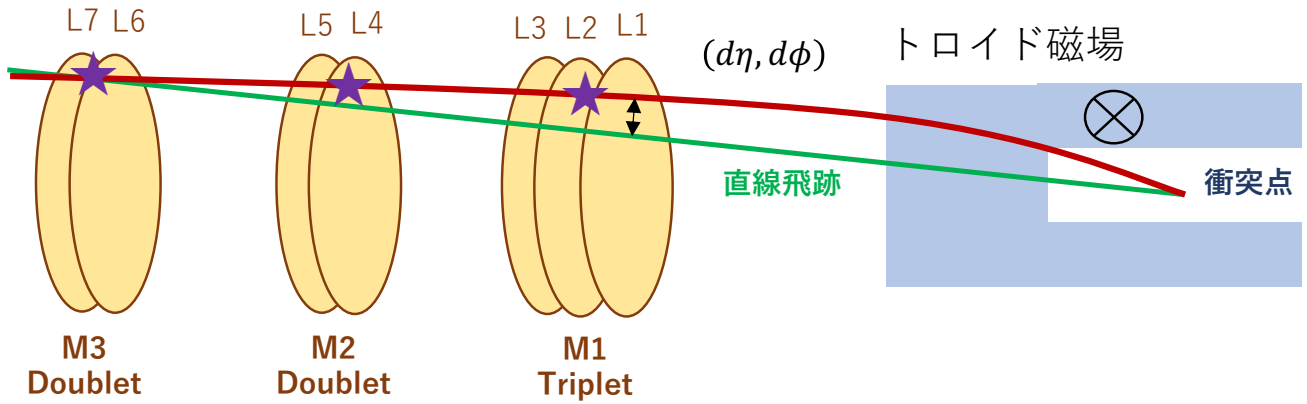


- SLトリガー系では多段の計算によってp_Tを得る
- Channel Mapping
  - トリガーロジックで扱いやすいように下準備をする
    - SLがシリアルリンクで受け取ったビットマップ (128 bit × 58 link) を コインシデンスのためのロジカルなチャンネル（各レイヤーで1次元配列）に変換する
  - この部分のみ、ファームウェアの開発も行った
- ステーション内コインシデンス
  - 各ステーションの代表点を得る
    - ロジカルなチャンネルに対して、AND、OR、NOTで記述されるコインシデンス論理をかける
- ステーション間コインシデンス
  - M1・M2・M3の代表点の組み合わせから、LUT(Look up table)によって無限運動量飛跡との差分$(\Delta\eta, \Delta\phi)$を得る
- Wire-Strip コインシデンス
  - $(\Delta\eta, \Delta\phi)$からLUTによってp_Tを得る

SLのトリガー系の7層コインシデンスの計算



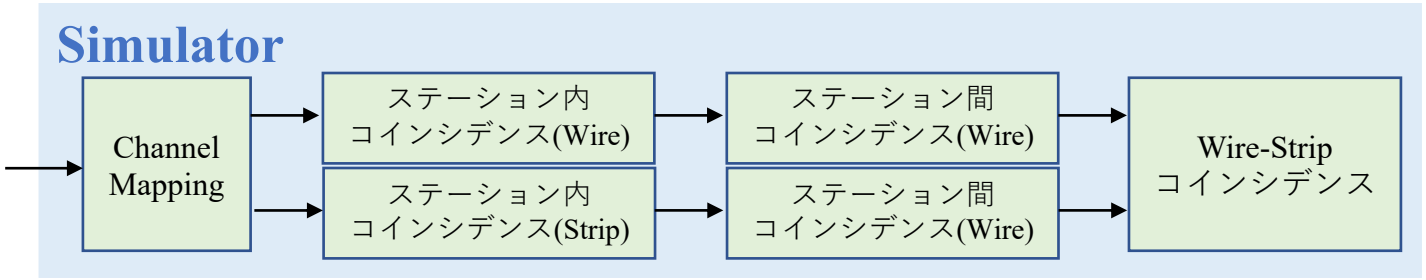


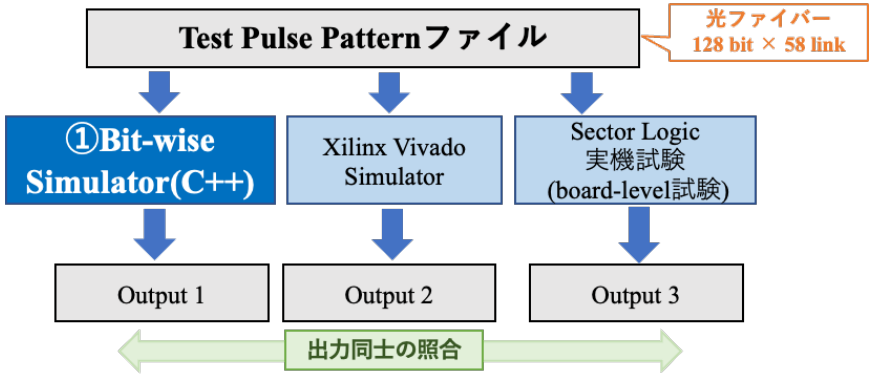


- SLトリガー系では多段の計算によってp_Tを得る
- Channel Mapping
  - トリガーロジックで扱いやすいように下準備をする
    - SLがシリアルリンクで受け取ったビットマップ (128 bit × 58 link) を コインシデンスのためのロジカルなチャンネル（各レイヤーで1次元配列）に変換する
  - この部分のみ、ファームウェアの開発も行った
- ステーション内コインシデンス
  - 各ステーションの代表点を得る
    - ロジカルなチャンネルに対して、AND、OR、NOTで記述されるコインシデンス論理をかける
- ステーション間コインシデンス
  - M1・M2・M3の代表点の組み合わせから、LUT(Look up table)によって無限運動量飛跡との差分$(\Delta\eta, \Delta\phi)$を得る
- Wire-Strip コインシデンス
  - $(\Delta\eta, \Delta\phi)$からLUTによってp_Tを得る

ビットワイズシミュレータの作成

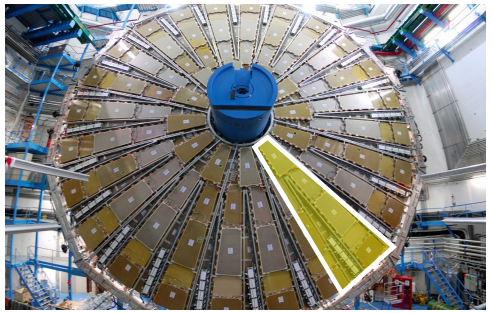
- 本研究ではファームウェアのロジックを忠実に再現したシミュレータをC++で作成した
- ファームウェアは多段のモジュールで構成され、シミュレータはモジュールの入出力の単位で完全一致
  - フロートによる近似ではなくビットによる計算を行う
    - 例：実機の信号線1本にbool値1つが対応
  - トリガーの制約に起因するファームウェアの構造も全て一致するようにソフトウェアを実装した
    - レイテンシーの制約から、全ての組み合わせではなく、優先順位をつけて上位いくつかのみをLUTに問い合わせる
    - 領域を分割し、独立に並列計算をすることで高速な計算を実現している
- 相互検証に使うため、ファームウェアと同一のテストパターンファイル・LUTのファイルを入力する
- ファームウェアシミュレータよりもイベントあたりの計算が高速
  - 多くのヒットパターンを試験する際に有用

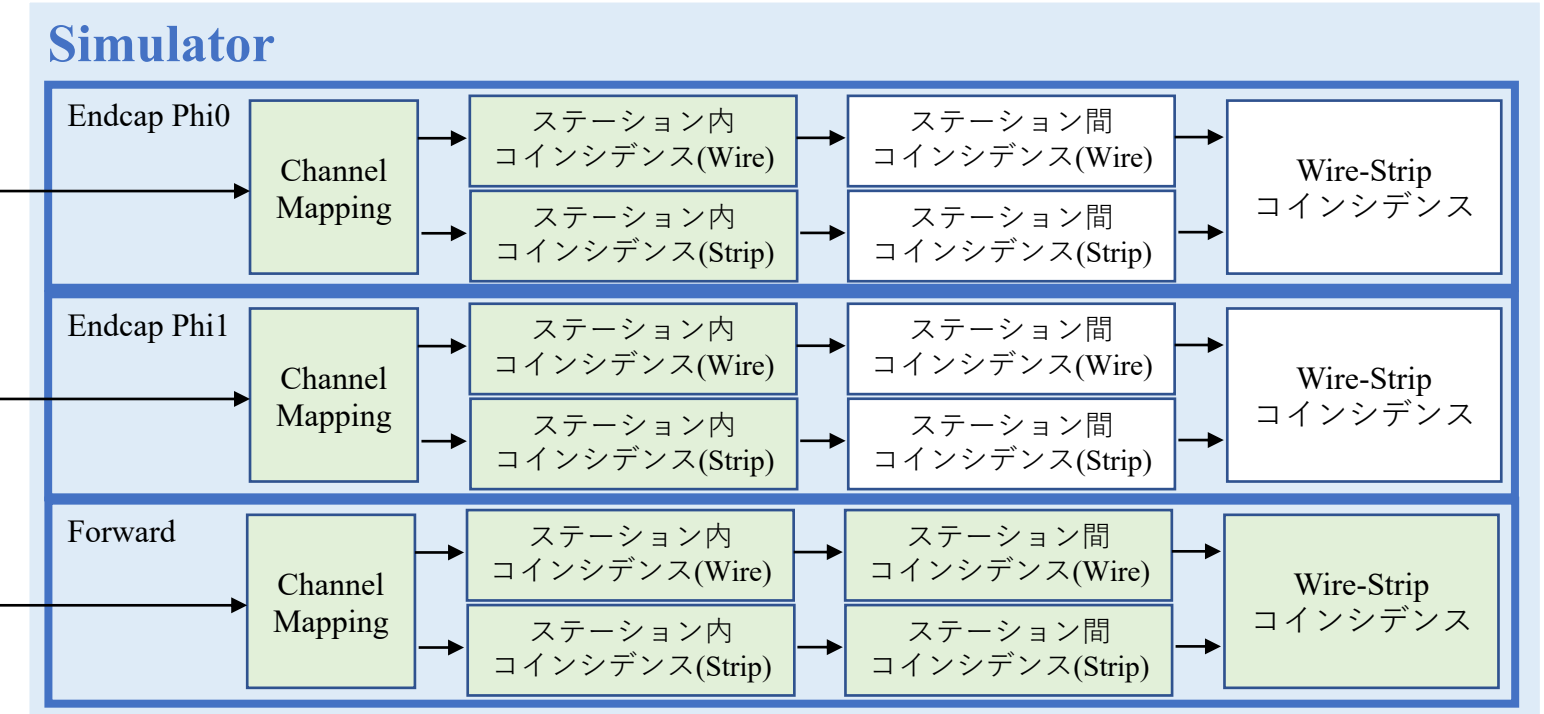


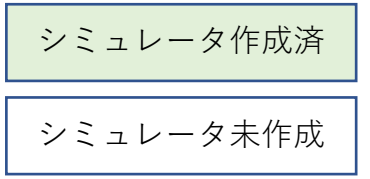


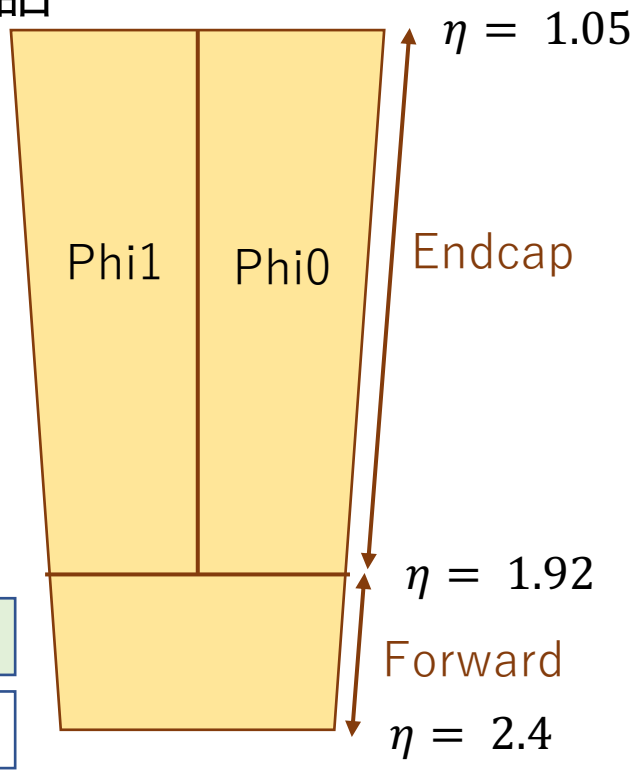
シミュレータ実装の現状

- 1枚のSLは1/24セクター（=“トリガーセクター”）を担う
- トリガーセクターは3つのサブセクターに分割され、ファームウェアはサブセクターごとに独立
  - フォワード x 1：$1.92 < |\eta| < 2.4$
  - エンドキャップ x 2： $1.05 < |\eta| < 1.92$
- 最後のセクションは図で示されるモジュールの検証を行なった話

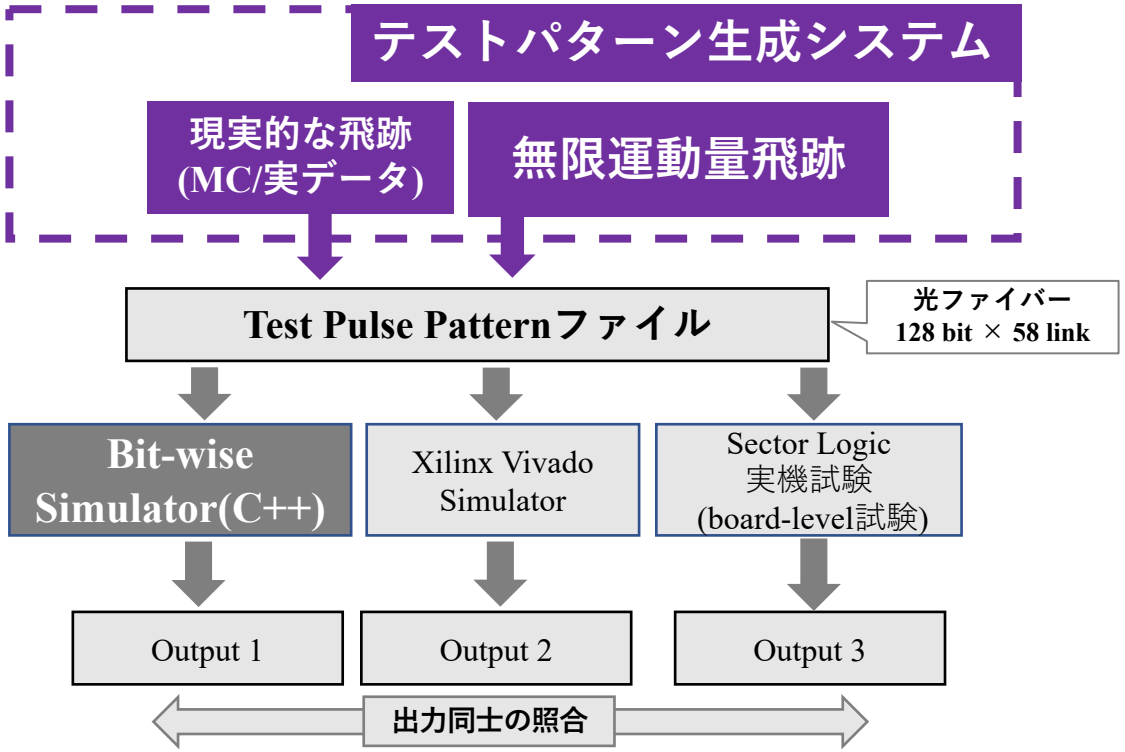




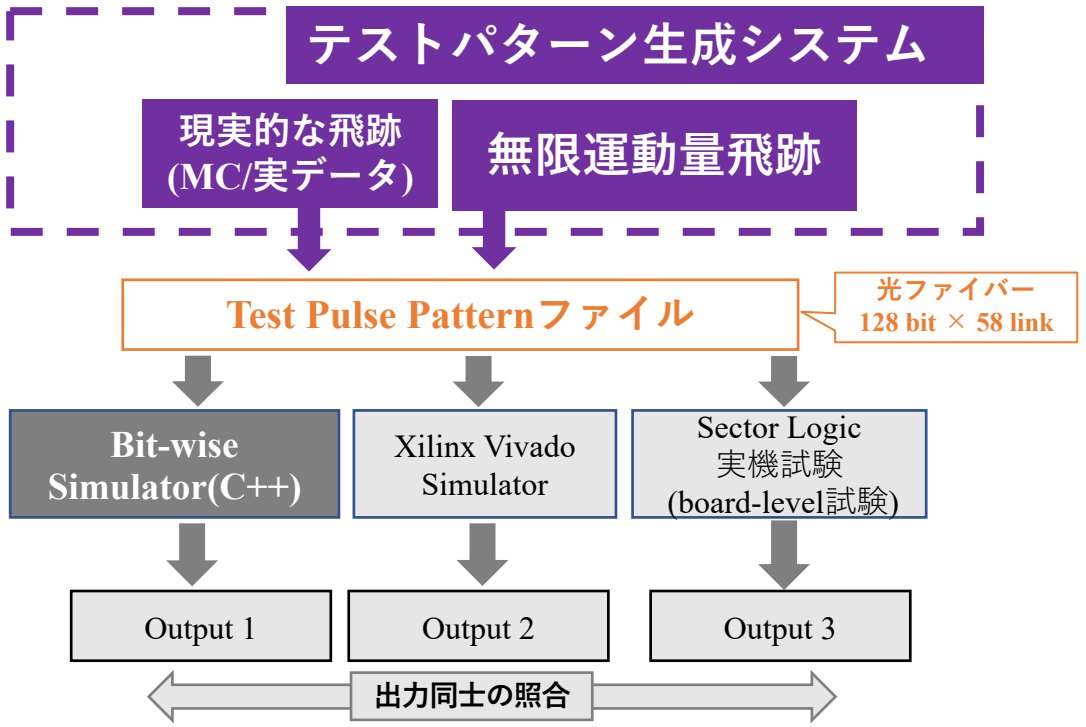


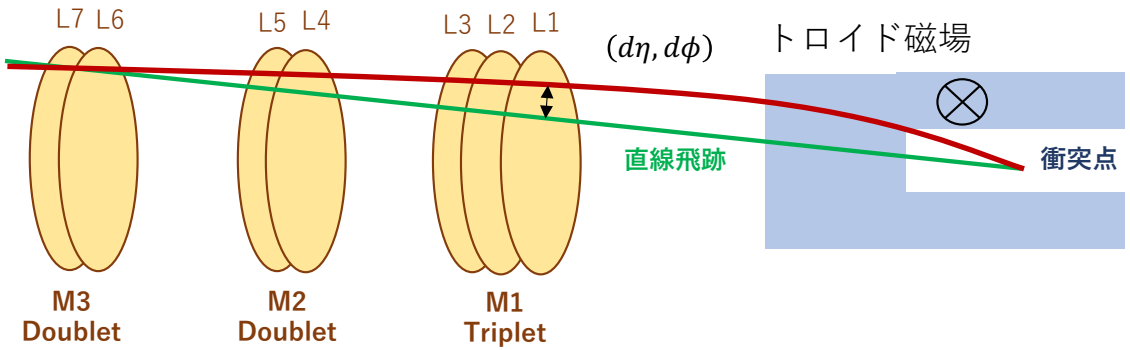


検証のためのテスト入力パターンの生成機構の開発



テストベクターの生成機構



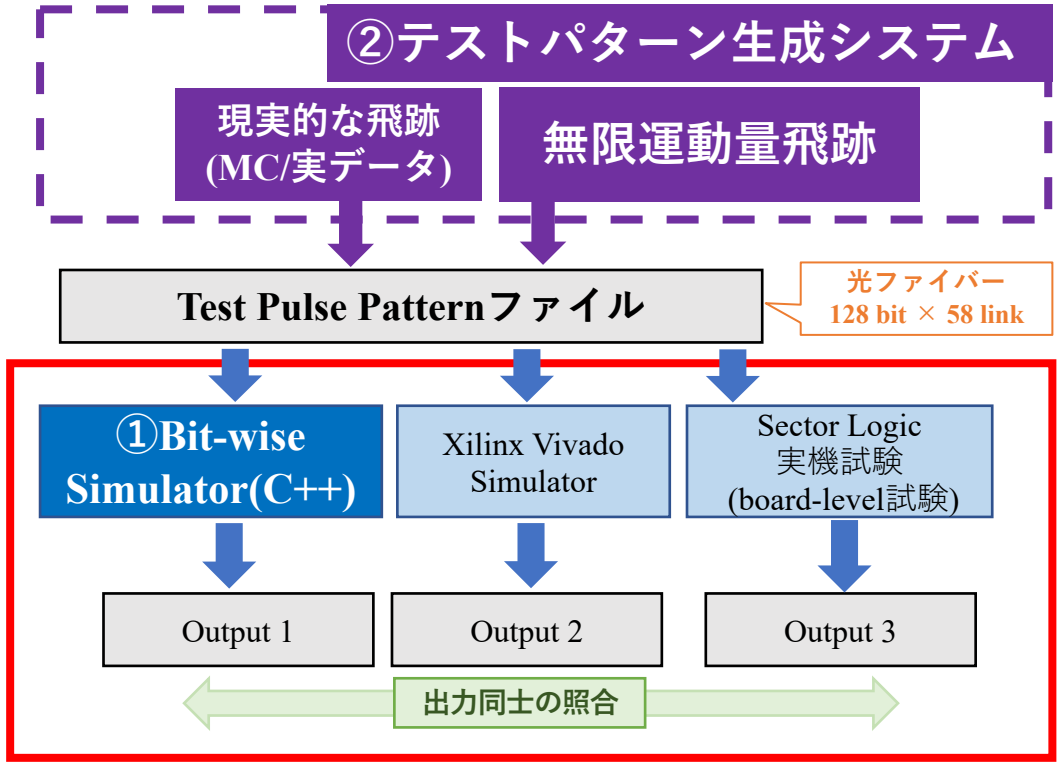


- トリガーロジックの検証のために、様々なテスト入力パターンを生成する機構を開発した
- 任意のヒットのパターンを、SLへの入力となる128bit × 58 link形式のテストベクターファイルに整形するための機構を作成した

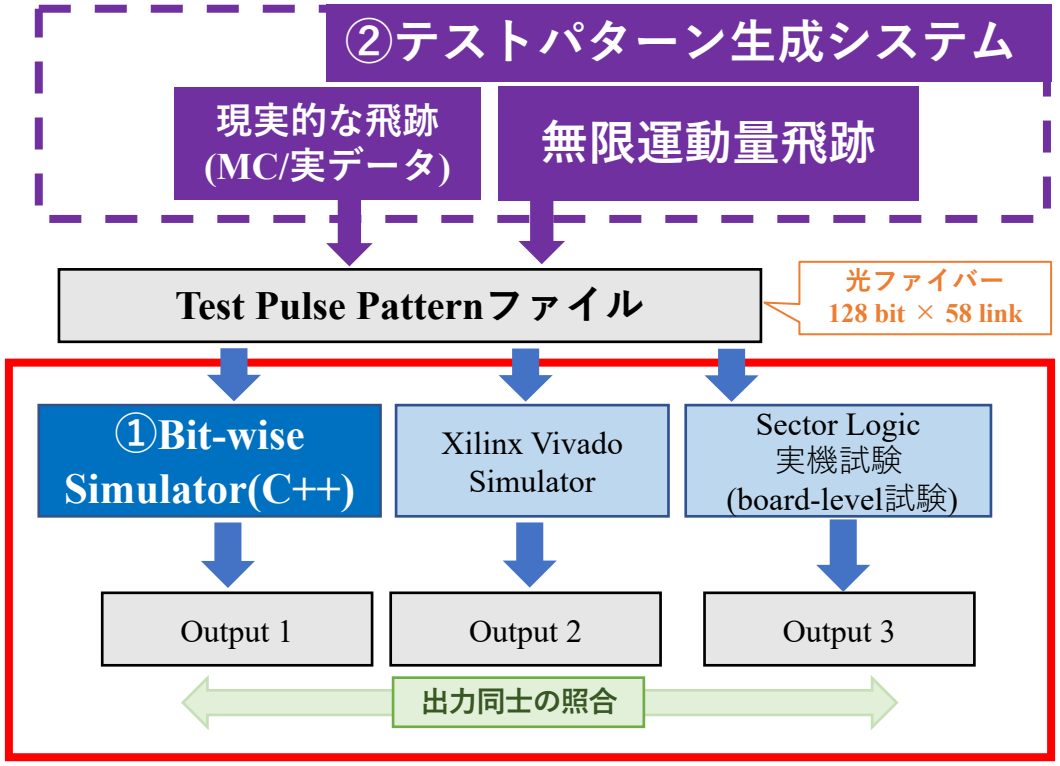
特に有用な2種類のテストベクターを実際に検証に使用

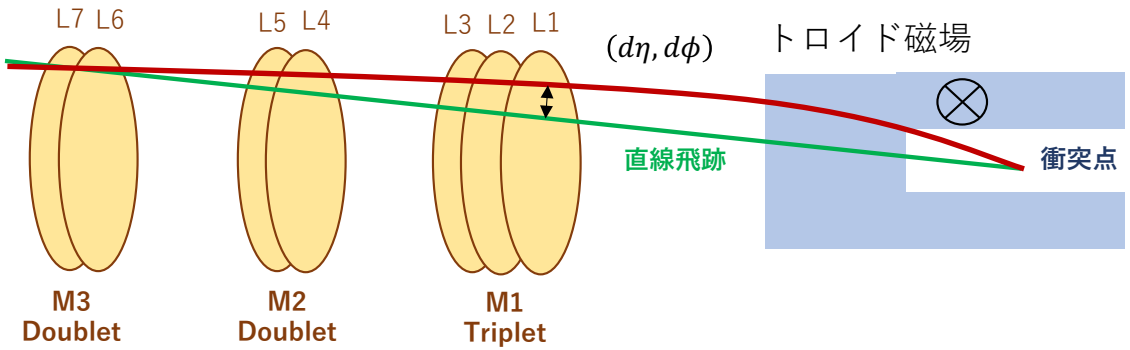
- **無限運動量飛跡**
  - 13層(Wire 7層・Strip 6層)突き抜けの直線飛跡
  - トリガーするべきトラックの中で最も単純かつ優先度の高いパターン
- **MCデータ/実データ**
  - 現実的な有限の曲率を持った飛跡による詳細なトリガーの検証

シミュレータとテストベクターを利用した検証



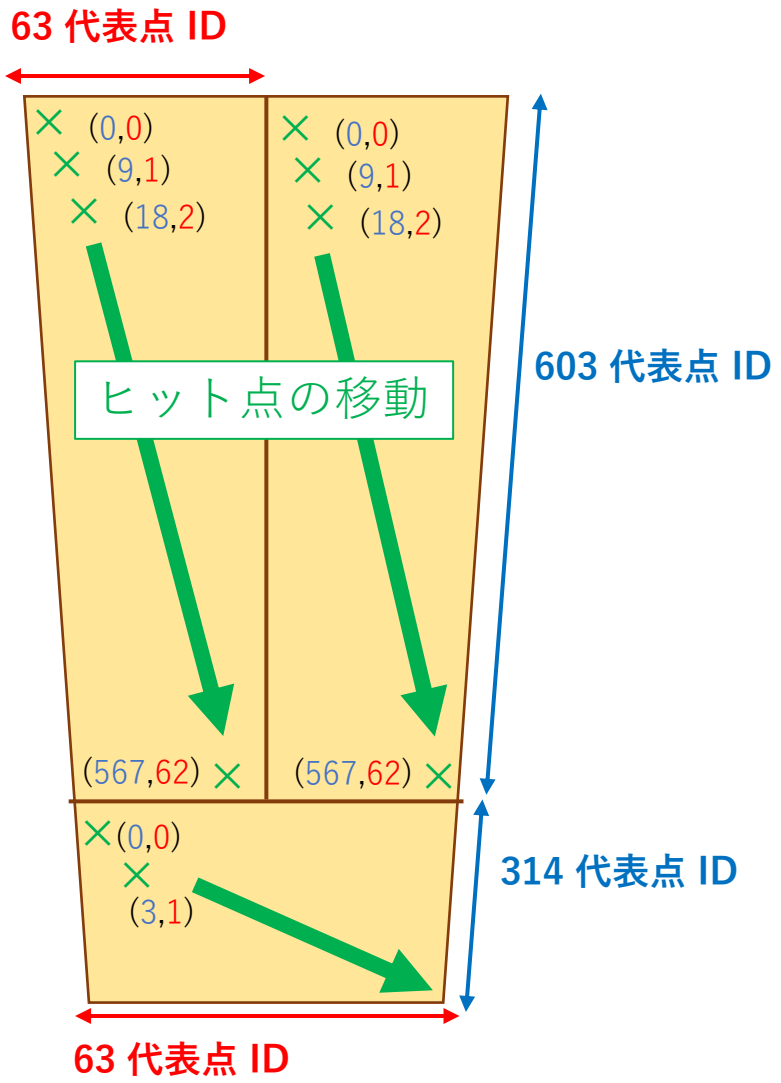
検証機構の実用



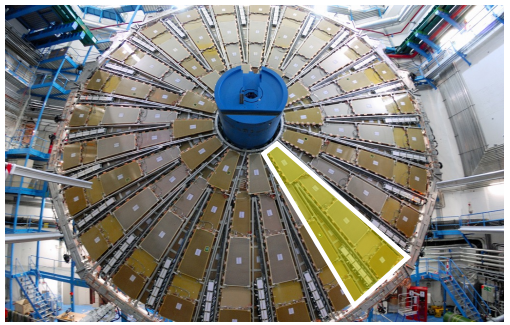


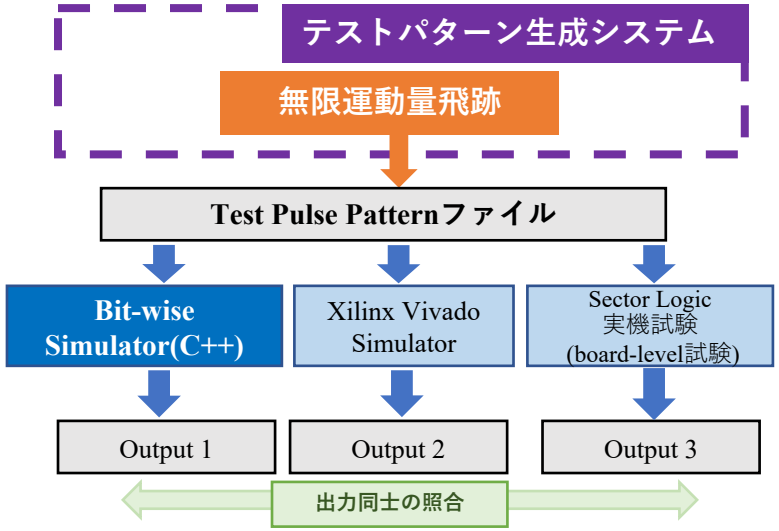
1. トリガーロジックの最初の検証として、無限運動量飛跡のテストベクターによる試験
   a. 無限運動量飛跡63イベントによる、実機・シミュレータ出力照合試験
   b. Forward Wireの範囲内にある直線飛跡全てを網羅したLUT検証試験
2. より詳細のトリガーロジック検証として、現実的な飛跡であるMCデータのテストベクターによる試験
3. 実機試験への活用

1-a：無限運動量飛跡63イベントによる出力照合試験

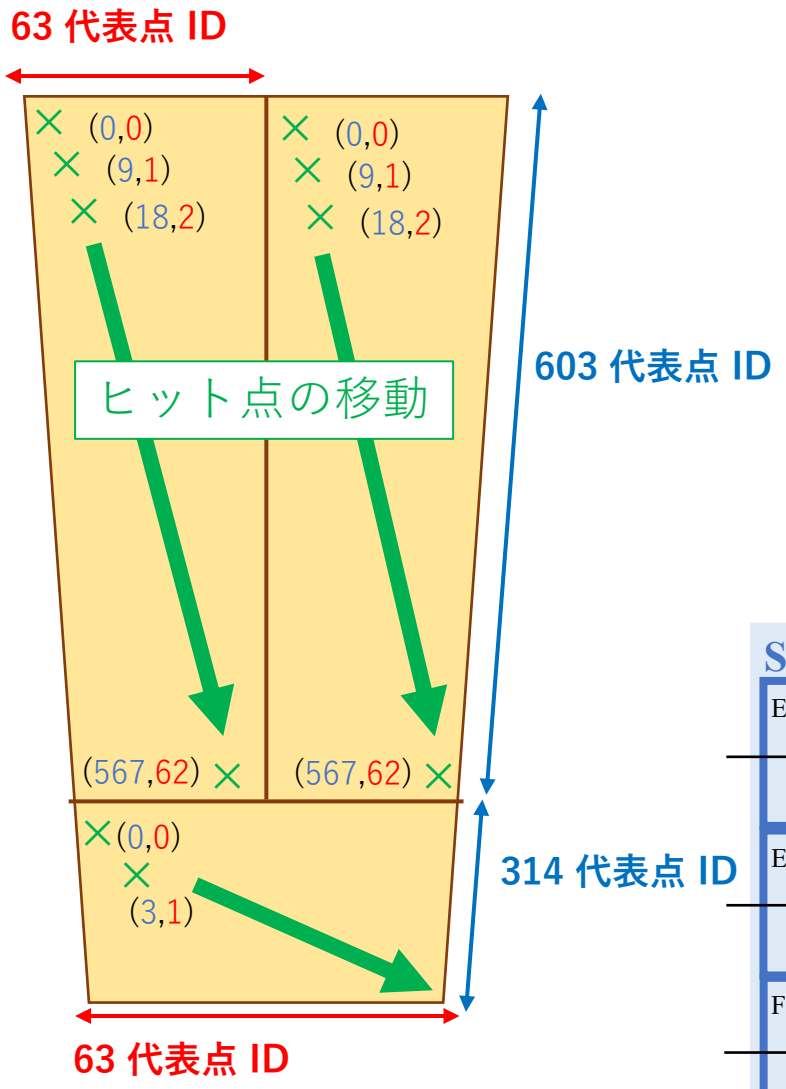


- 無限運動量飛跡63イベントを
  セクターロジック実機、
  ファームウェアシミュレータ(Vivado)、
  ビットワイズシミュレータ
  の3つのパスに入力し、出力を照合した
  - サブセクターごとに無限運動量飛跡1本を構成する13点(Wire 7層+Strip 6層)を入力
  - セクター内を満遍なく検証するため、チェンバーを斜めに横断するように、イベントごとにヒットの位置を移動



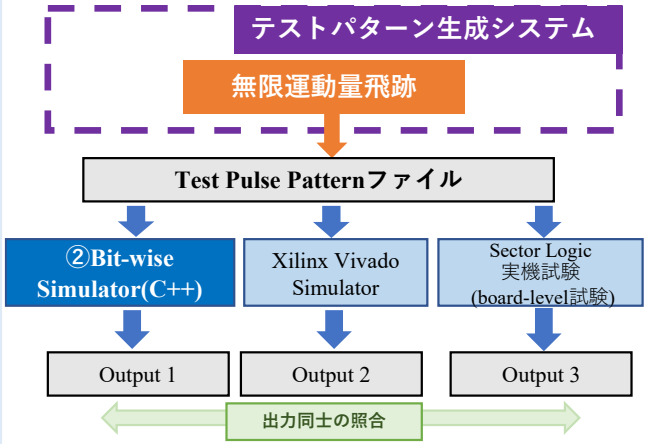


1-a：無限運動量飛跡63イベントによる出力照合試験

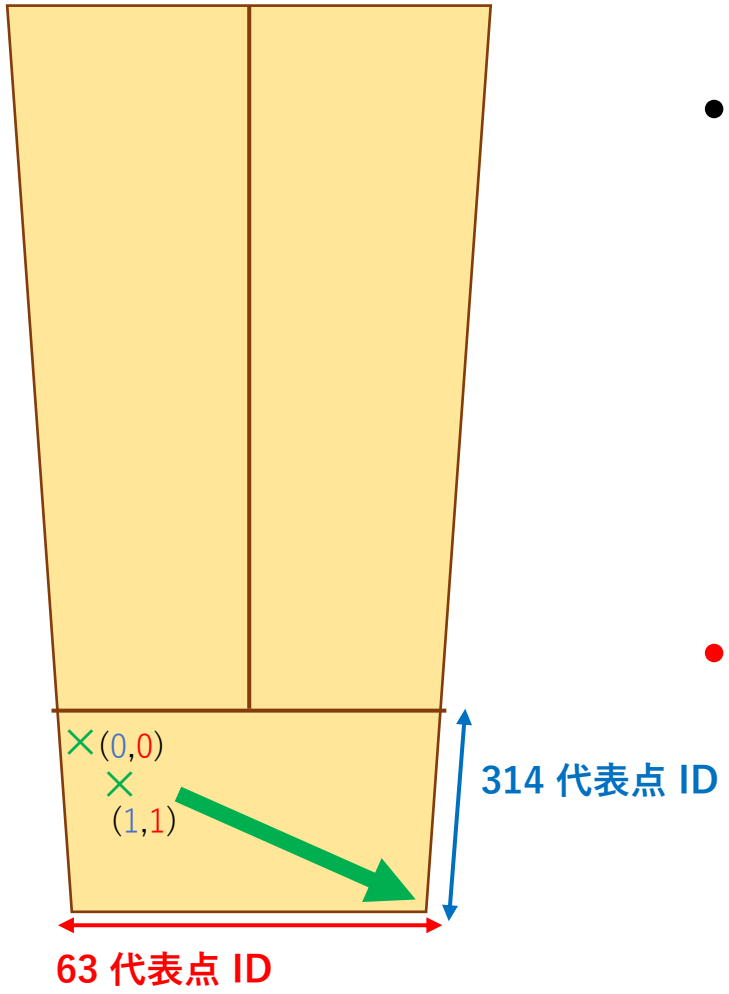


検証結果

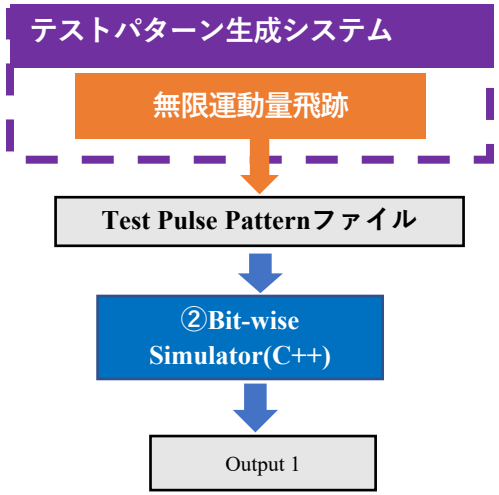
- Endcap
  - ~ステーション内コインシデンス：完全に一致
- Forward
  - ~ステーション内コインシデンス：完全に一致
  - ステーション間コインシデンス(Wire)：完全に一致
  - ステーション間コインシデンス(Strip)：62/63イベント一致
    - ファームウェアシミュレータで再構成できず、ビットワイズシミュレータでは再構成できた
    - 原因を調査中



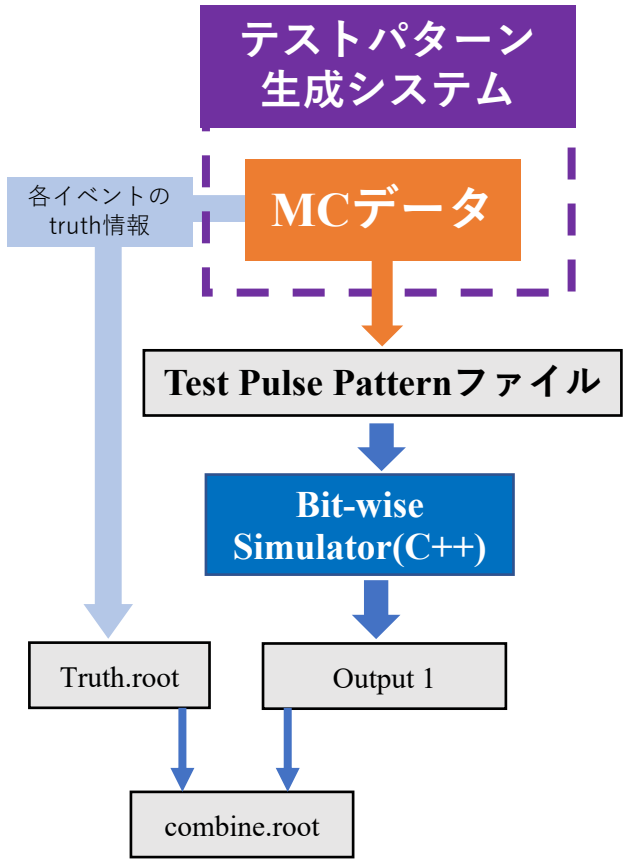
1-b：無限運動量飛跡によるFWのLUT検証試験



- 目的
  - ステーション間コインシデンス(Wire)のLUTの検証
- 検証手法
  - ビットワイズシミュレータ
  - テストベクター
    - Forward Wire内の代表点242点への無限運動量飛跡
      - 242点：M1～M3にチャンネルが存在する無限運動量飛跡
      - 無限運動量飛跡＝確実にトリガーすべきパターン
    - 1イベントにつき1トラックで242イベント
- 検証結果
  - 再構成できないパターンが2つ見つかった
  - LUTを作成している共同研究者に
    フィードバックし、LUTが修正された



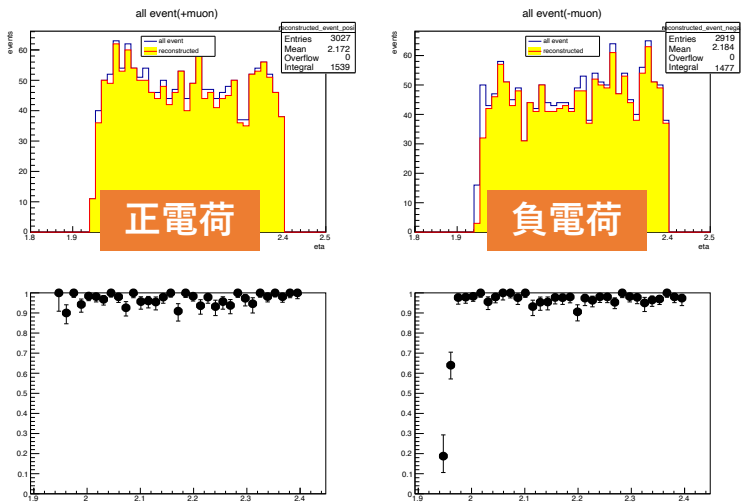
2：MCデータによるefficiencyの検証試験



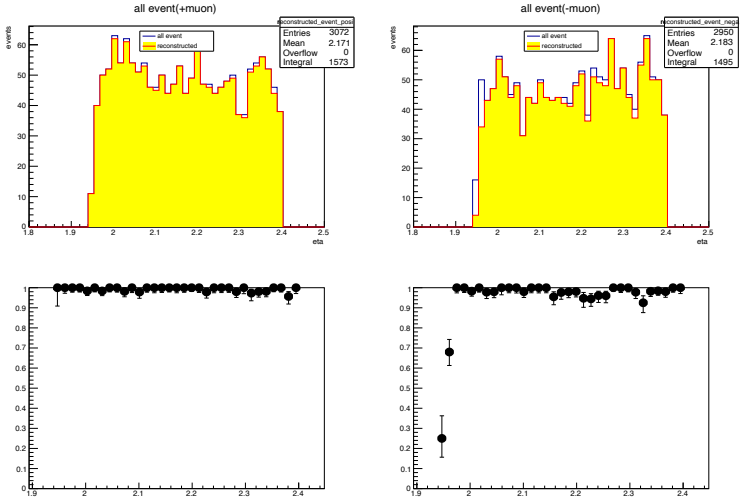
- 目的
  - 大統計・現実的な入力に対するビットレベルのトリガーアルゴリズムの応答を見て性能評価をする
- 手法
  - MCデータから生成したテストベクターをビットワイズシミュレータに入力
  - Truth情報(ミューオンの(η, ϕ)座標、p_T)と各モジュールの出力情報を用いて解析

2：MCデータによるefficiencyの検証試験

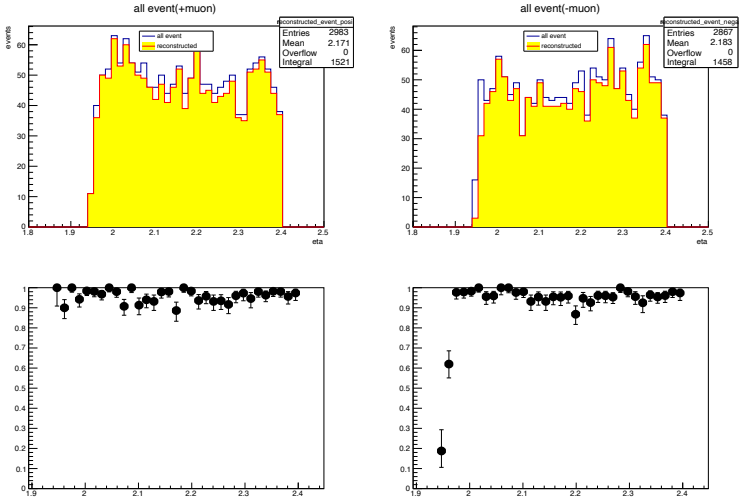
- 6000 eventの試験を行なった
  - $p_T > 40$ GeV（確実にトリガーすべきイベント）
  - ミューオンが1本のみのイベント（初期の試験なので簡単のため）
  - $1.95 < \eta < 2.4,\ 2.87 < \phi < 3.14$(Forward 領域のうち、TGC検出器の構造によるinefficiencyがない領域)
- LUTから何らかの値を取り出せたイベントの割合をそれぞれ確認した(下図)
  - 横軸η
  - 期待通りのefficiencyが出力された



ステーション間コインシデンス(Wire)
平均Efficiency = 0.97



ステーション間コインシデンス(Strip)
平均Efficiency = 0.98



Wire-Strip コインシデンス
平均Efficiency = 0.95

イベントの解析：Wire-Strip コインシデンス

- 再構成できなかったイベントの調査を行った

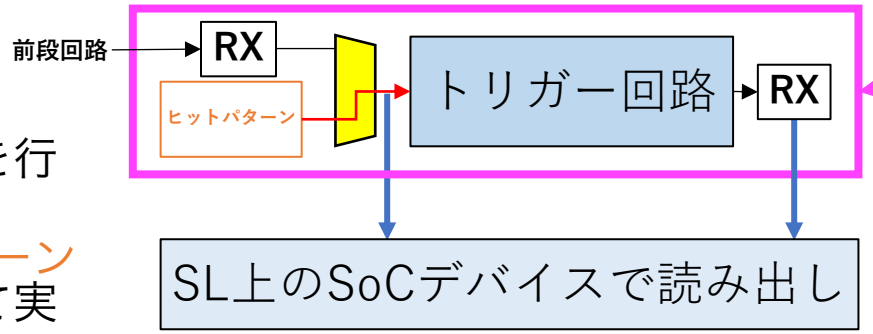
	イベントの分類	イベント数 （全6000イベント）
LUT内にデータなし	Wire, Strip両方のステーション間コインシデンスから入力あり	39
LUTへの入力が不完全	Stripからのみ入力あり	133
	Wireからのみ入力あり	57
	どちらからも入力なし	73

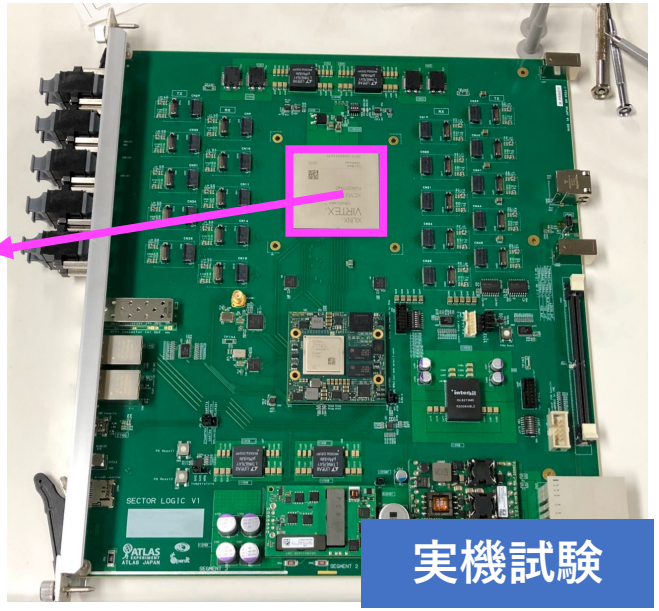
  - LUTにデータがなかったイベントについて、inefficiency理屈をイベントごとに調査
    - Stripのステーション間コインシデンスの出力となる$d\phi$（内部磁場による飛跡の歪曲具合）が大きい
    - ノイズのないMCデータのイベントのみを用いているのに、ステーションあたりの代表点が多い（約10個～）
    - 大量の飛跡候補の中から、直線に近い飛跡を選択できていない
  - 物理的な解釈
    - MCデータのTruthの情報を簡単に辿れないので推測になるが、ミューオンと検出器等の物質との相互作用で散乱している多重散乱事象だろうと推測される
  - 飛跡選別のロジックの確認
    - ファームウェア自体のロジックによるinefficiencyなのか、シミュレータ開発時にロジックを誤翻訳したためのinefficiencyなのかを確認する必要がある
    - これから、該当するイベントのみを集めてテストパターンセットを作成し、ファームウェア側でも試験を行う（次ページ）
- このように、シミュレータの各モジュールの出力も全て記録しているため、予期せぬinefficiencyなどについて簡単に解析をすることができる

3：実機試験への活用

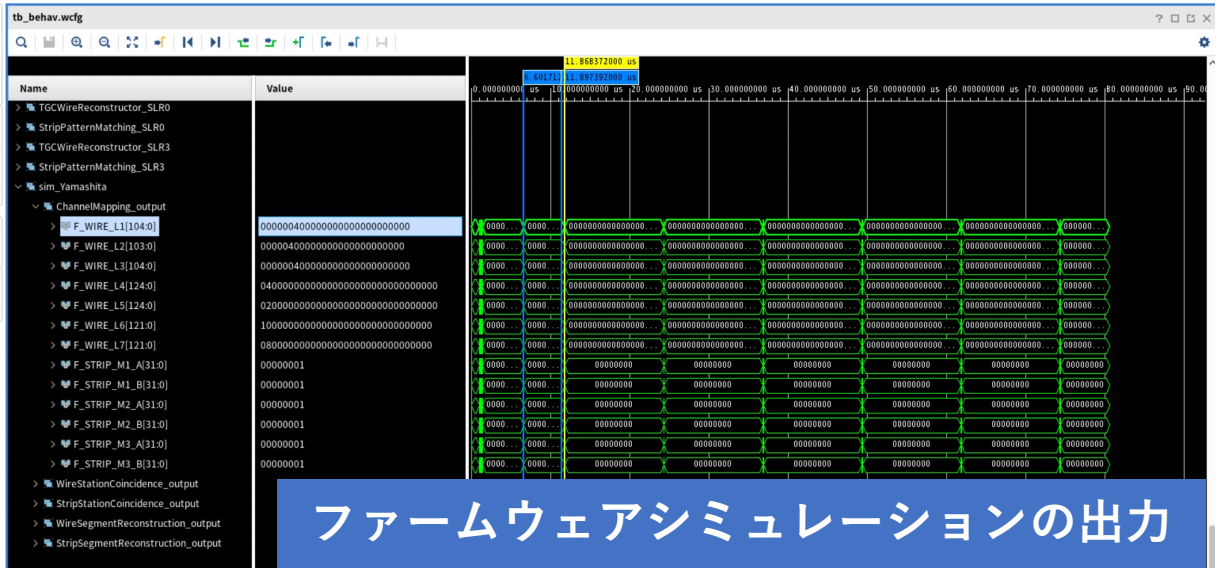
- 実機の試験とファームウェアの試験に同じ入力を使用し、相互検証

- 実機試験
  - 実機ではボード単体でのトリガー回路の試験フレームワークを実現し、動作検証を行なっている
  - ファームウェア内のRAMにトラックパターンファイルを仕込んで、任意の入力に対して実機の応答を確認することができる
- 出力の比較
  - 本研究の枠組みを使うと、同一のテストパターンによる実機試験、ファームウェアシミュレーション、ソフトウェアシミュレーションの相互照合を容易に行うことができる
  - SLのファームウェアのIO、読み出し、トリガー、制御の全回路の統合が共同研究者(同大学の三島)によって行われたが、その際の検証に活用されている
  - 今後さらに多様な入力による実機、ファームウェアシミュレータ、ソフトウェアシミュレータの相互照合をしながら開発・運用ができる環境が整った



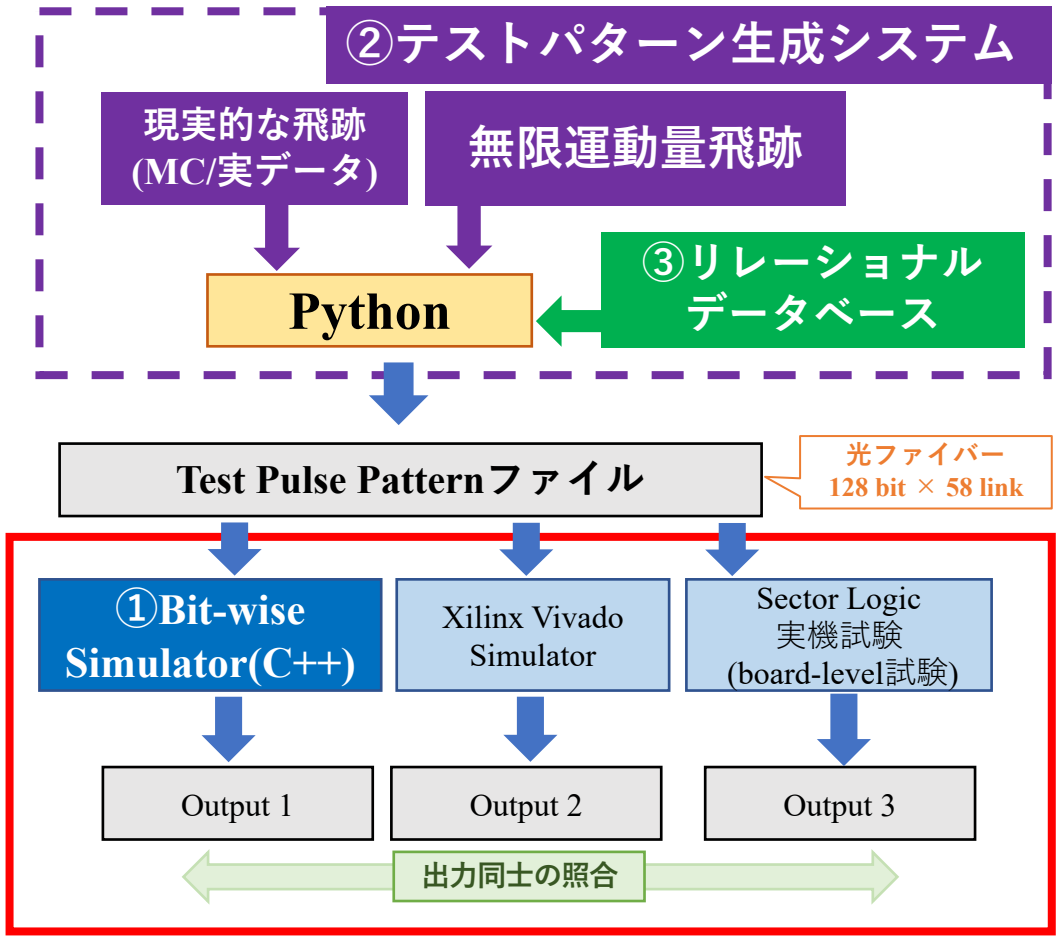


実機試験



ファームウェアシミュレーションの出力

まとめ



- Phase2に向けた開発が進んでいる
- それに向けた開発フレームワークとしてSLの検証機構の全体設計を行い、その構成要素を開発した
  - ビットワイズシミュレータ
  - テストパターン生成システム
  - リレーショナルデータベース
- 実際に検証機構を運用し、SL開発研究に役立てている

将来の展望

- 本検証機構は現在のSL開発研究のみならず、SL実機の運用時も見据えた設計となっている。
  - 例：運転中、試運転中のファームウェアの最初の検証や、システムの診断

バックアップ

イベントの解析：ステーション間コインシデンス(Wire)

- 再構成できなかったイベントの精査を行った
  - Wire/Stripの比較
    - 「検出器自体にヒットはあったが、 Segment Reconstructionの段階で落ちている」イベントがStripと比較して多い
      (Wire : Strip=145 event : 71 event, Wireのinefficiencyの70%)
  - inefficiency理屈をイベントごとに調査
    - 若干の折れ曲がりをもった飛跡（LUTに存在しない）
    - 現在のトリガーでは落ちるべくして落ちている事象であり、Inefficiencyのメカニズムをクリアに理解できた
  - 若干の折れ曲がりに対する物理的な解釈
    - Truthの情報を簡単に辿れないので推測になるが、ミューオンと検出器等の物質との相互作用で散乱している多重散乱事象だろうと推測される
    - 多重散乱事象に対応するデータがLUT内に用意されていないためのinefficiencyと結論付けられるが、原理的に回復可能なものなので、よく理解をするべき点である

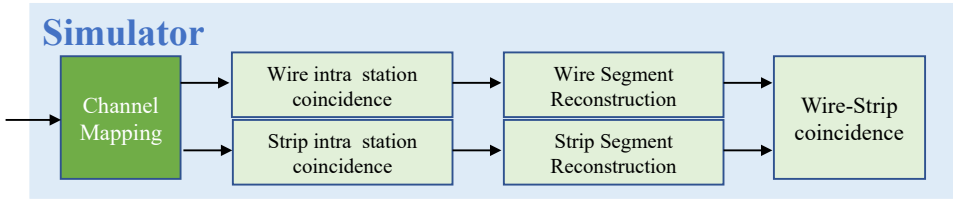
再構成できなかったイベントの例

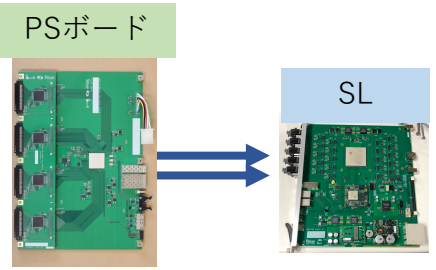
M1	M2	M3
224	224	224
225	225	225
226	226	226
227	227	227
228	228	228

同じηのチャンネルが同じ番号になる
スタッガードチャンネルによる表記
（同じチャンネル番号の組＝無限運動量飛跡）

Channel Mapping

- Channel Mapping
  - L0トリガー系のファームウェアで最初のモジュール
  - SLがシリアルリンクで受け取ったビットマップ(128 bit × 58 link) を コインシデンスのためのロジカルなチャンネルに変換する
    - SLは29枚のPSボードから58本の光ファイバーによりTGC Big Wheelのヒット情報を受け取る
    - 1本の光ファイバーが最大128チャンネルを取り扱う
  - 出力はWired ORを考慮した「トリガー系入力チャンネル」





- このモジュールに限り、この研究でシミュレータだけでなくファームウェアも開発した
  - データベースを利用してファームウェアを自動生成する仕掛けを開発した
  - ビットポジションとロジカルなチャンネルの対応関係をデータベースから取り出し、Pythonでファームウェアの形に整形した
  - SL 1枚が担う領域(トリガーセクター)にある約6000のチャンネルについて、1チャンネルにつき1行でファームウェアが記述される

出力： レイヤー & チャンネル番号　　入力：リンク番号 & ビットポジション

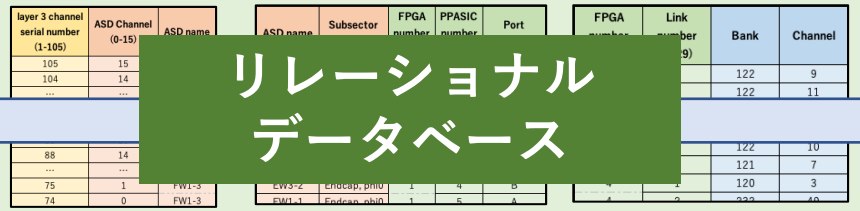
Python

ファームウェアの抜粋

```
OUTPUT_ENDCAP_PHI0_WIRE_L3(148) <= INPUT_BITMAPS_E_phi0(6)(9);
OUTPUT_ENDCAP_PHI0_WIRE_L3(149) <= INPUT_BITMAPS_E_phi0(6)(10);
OUTPUT_ENDCAP_PHI0_WIRE_L3(150) <= INPUT_BITMAPS_E_phi0(6)(11);
OUTPUT_ENDCAP_PHI0_WIRE_L3(151) <= INPUT_BITMAPS_E_phi0(6)(12);
OUTPUT_ENDCAP_PHI0_WIRE_L3(152) <= ( INPUT_BITMAPS_E_phi0(6)(13) or INPUT_BITMAPS_E_phi0(11)(21) );
OUTPUT_ENDCAP_PHI0_WIRE_L3(153) <= INPUT_BITMAPS_E_phi0(11)(22);
```

3 サブセクター × 13 レイヤー ×チャンネル番号
(Wire 7 レイヤー, Strip 6 レイヤー)

TGC検出器のチャンネルID（コインシデンスロジックへの入力）



リレーショナルデータベース

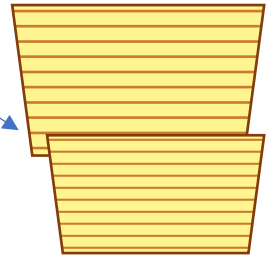
光ファイバー
128 bit × 58 link

SLへのシリアルリンクのデータ形式のビットポジション

Wired ORを考慮したレイヤー毎のチャンネル番号

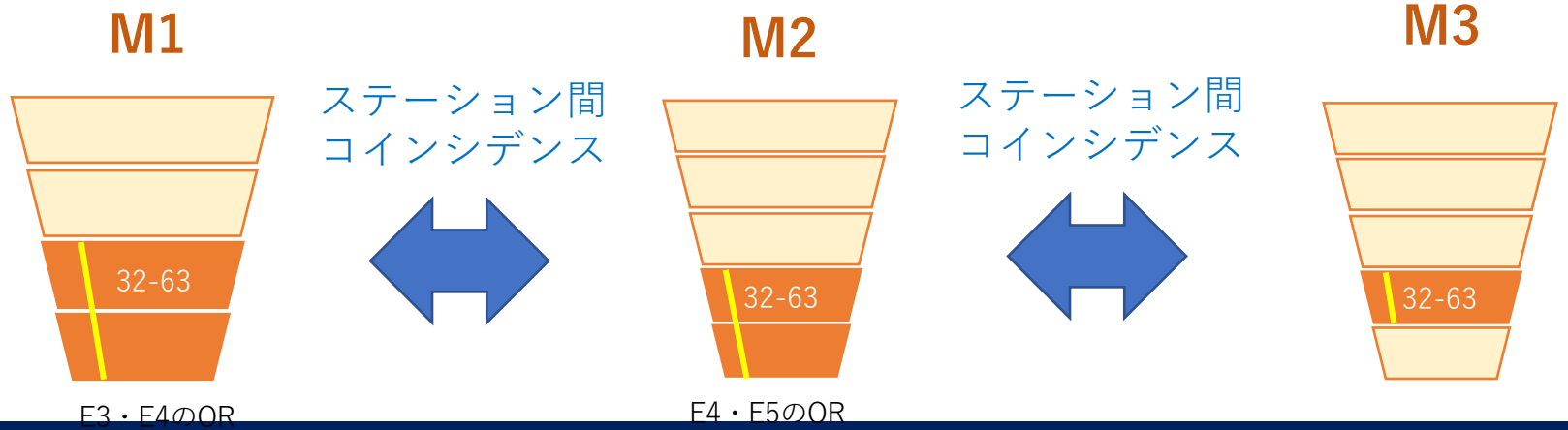
- ステーション内コインシデンスを行うために、各レイヤーのヒット情報を1次元配列に整形する
  - トリガーセクターは複数のチェンバーで構成されるので、トリガー計算をするときはチェンバー構造を考慮する必要がある
  - トリガー系の最初でチェンバー構造の複雑さを吸収したチャンネルに整形することで、以降の論理回路では構造を気にせずに演算できるようにした
  - この配列を定義した。ここでは「トリガー系入力チャンネル」と呼ぶ ※修士論文本文での名称は「ステーション内コインシデンス入力」
- ワイヤーにおけるWired OR
  - チェンバー境界のオーバーラップしたチャンネルに対するOR
  - 「トリガー系入力チャンネル」は現行システムにおけるレイヤー内通し番号を踏襲した

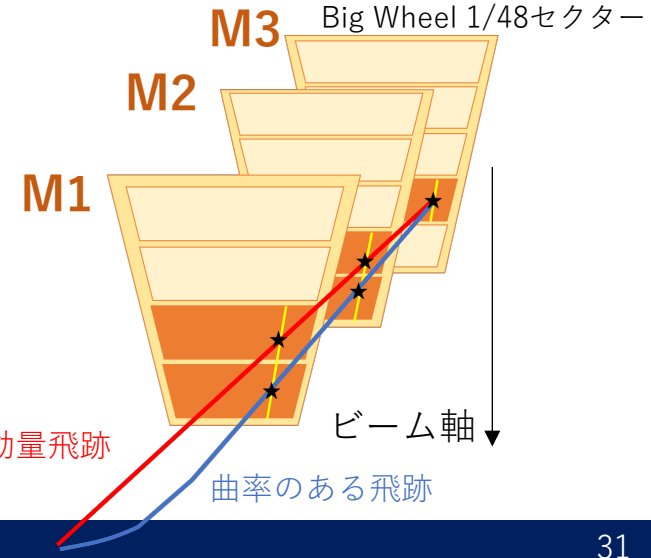
Wired OR
チェンバー境界に同じη領域を観測するチャンネルがある



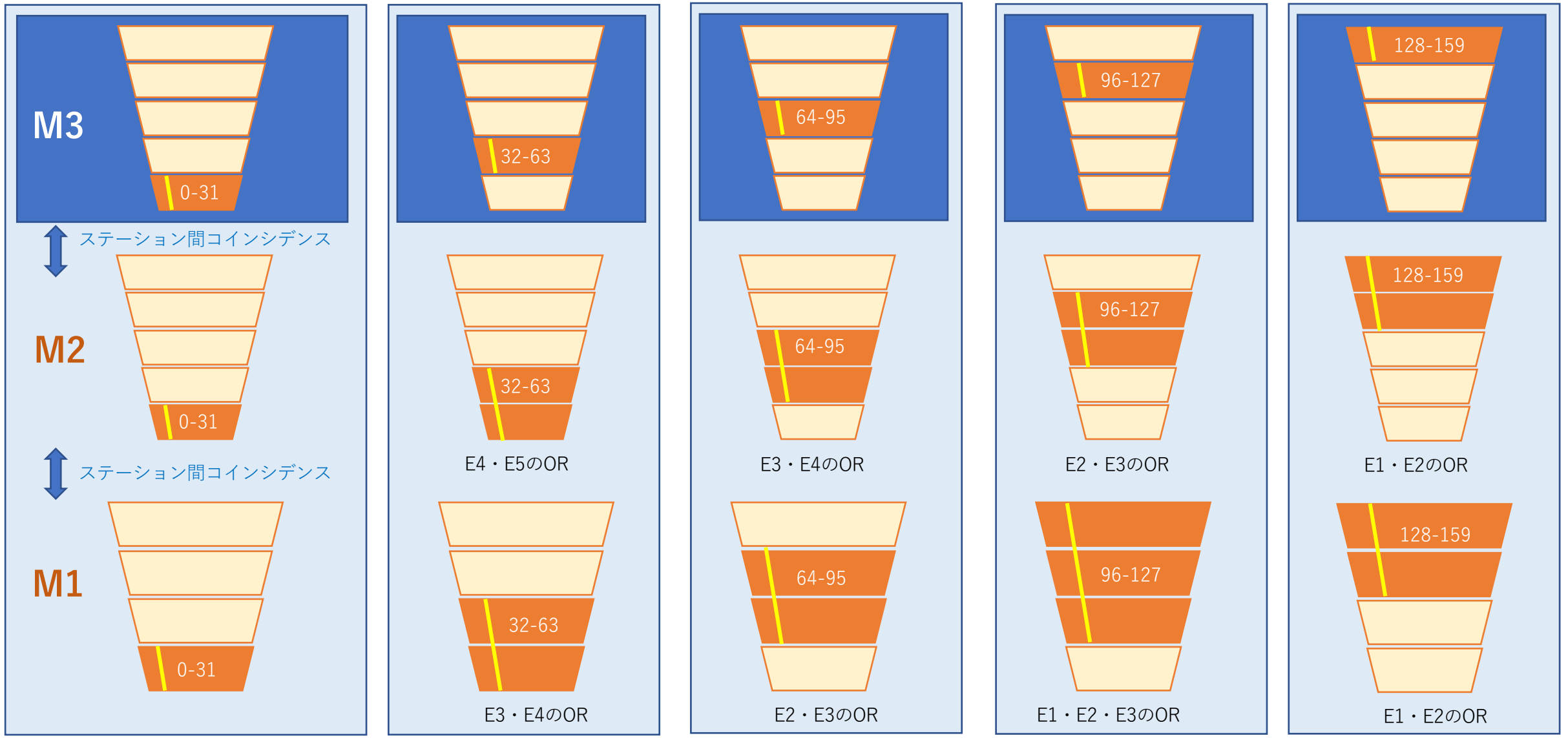
- ストリップにおけるWired OR
  - ステーション間でのコインシデンスでは、M3がピボットとなる
  - M1・M2 ステーションでは複数のチェンバーのチャンネル同士でOR を取り、チェンバーをまたがる仮想的な長いチャンネルとして取り扱う必要がある
    - 理由1：ステーション間でのチェンバー境界のη 座標が一致しない
    - 理由2：内部のトロイド磁場により飛跡がη 方向に曲がる
  - M3のピボットを元に、チェンバーをまたいでORをとった仮想的な長いチャンネルを「トリガー系入力チャンネル」と定義した

ピボットとなるM3のチェンバーに対してコインシデンスを取るべき領域

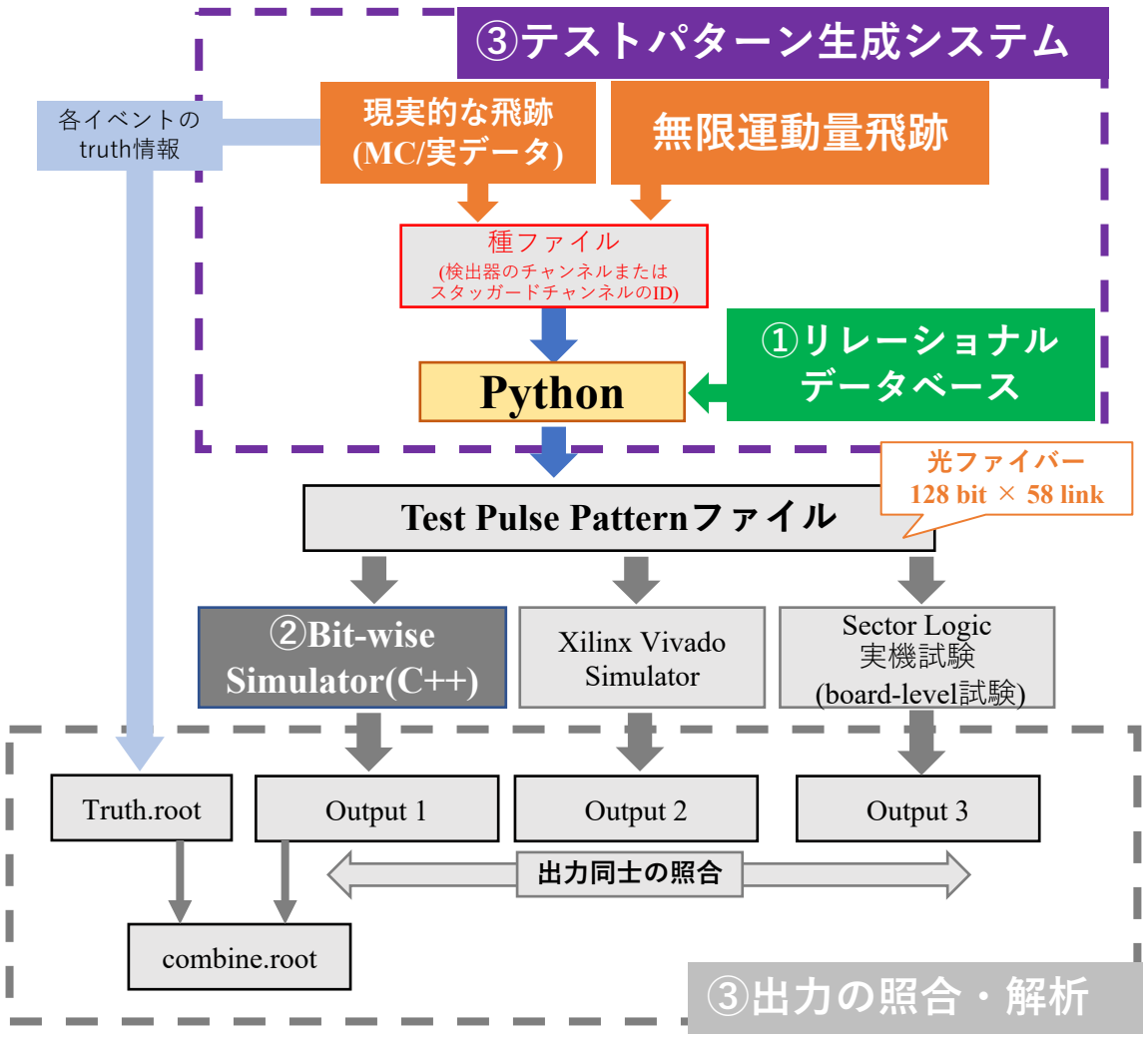




ストリップのトリガー系入力チャンネル

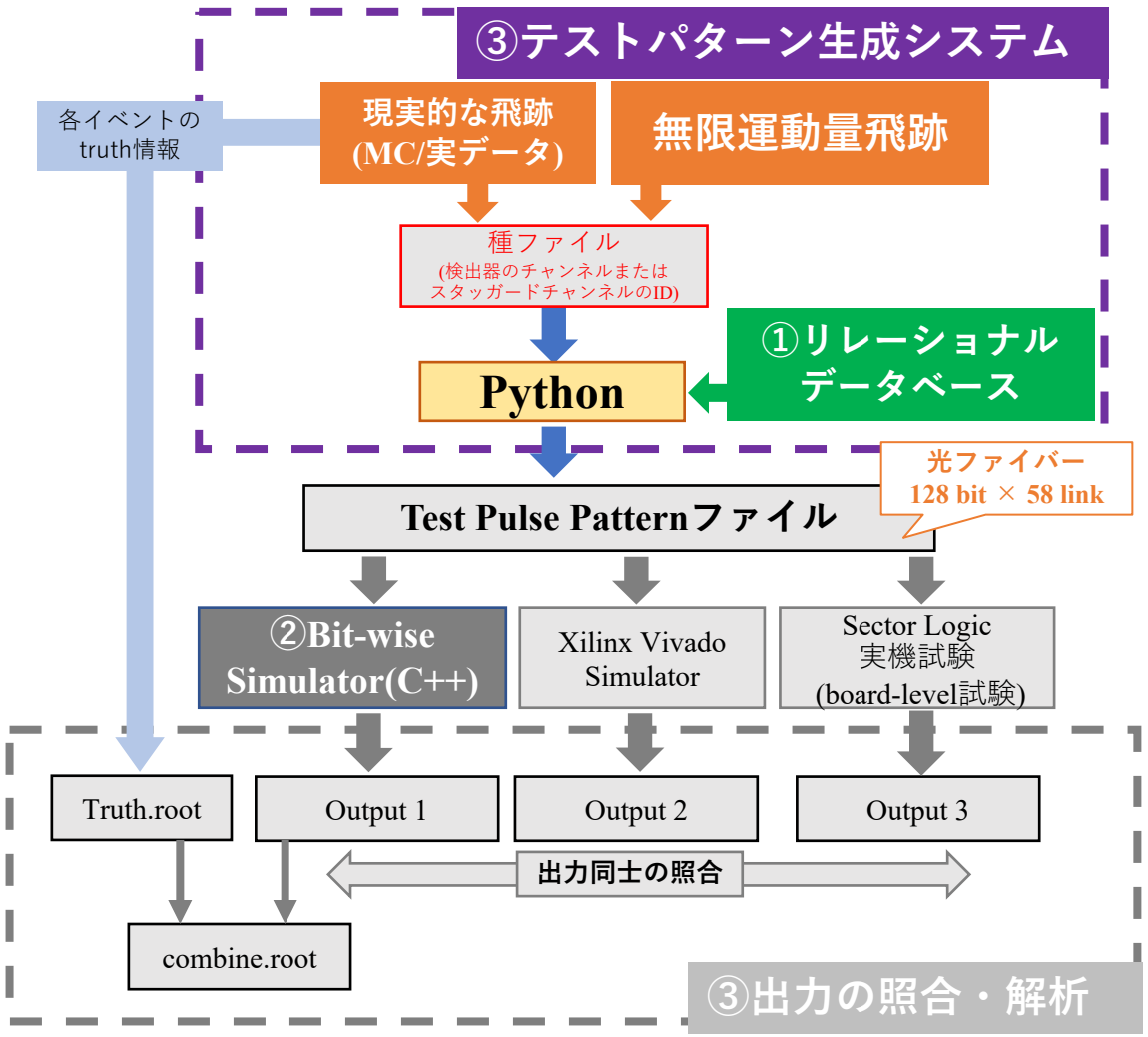


テストベクターの生成機構



- 目的：トリガーロジックの検証のために、様々なテスト入力パターンを生成する
- 任意のヒットのパターンを、SLへの入力となる128bit × 58 link形式のテストベクターファイルに整形するための機構を作成した
  - 入力1：ヒットを打つチャンネルを指定する種ファイル
  - 入力2：リレーショナル・データベース
- 種ファイル：柔軟な入力が可能
  - 1つのファイルに任意のイベント数を書き込める
  - 複数のフォーマットを用意
    - スタッガードチャンネルを入力し、無限運動量飛跡となる13点(Wire 7層+Strip 6層)をテストベクターに書き込む
    - 検出器上のチャンネルを1点ずつ指定
    - ランダムなノイズを指定された数打つ
  - 1イベントに対して複数のヒット点を指定することが可能
    - （例）無限運動量飛跡 x2 ＋ノイズ
  - これらのフォーマットを組み合わせることで様々な検証が可能になる

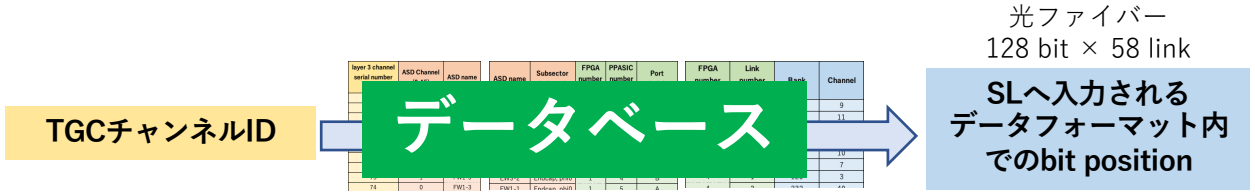
テストベクターの生成機構



特に有用な2種類のテストベクター (実際に検証に使用)

- 無限運動量飛跡
  - トリガーするべきトラックの中で
    最も単純かつ優先度の高いパターン
  - スタッガードチャンネル番号を使用することで実現
    - 13層(Wire 7層・Strip 6層)突き抜けのトラック

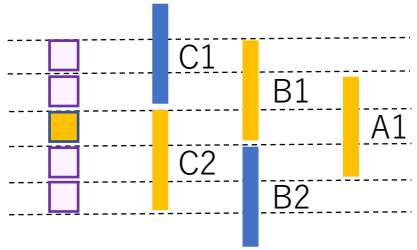
- MC データ/実データ
  - 現実的な有限の曲率を持った飛跡による
    詳細なトリガーの検証
  - MC/実データのファイル(ROOTファイル)内のヒット点情報に従って1点ずつチャンネルを指定することで実現
    - MCデータではミューオンの四元運動量の情報なども入っているので、イベントを任意の条件で選別することもできる

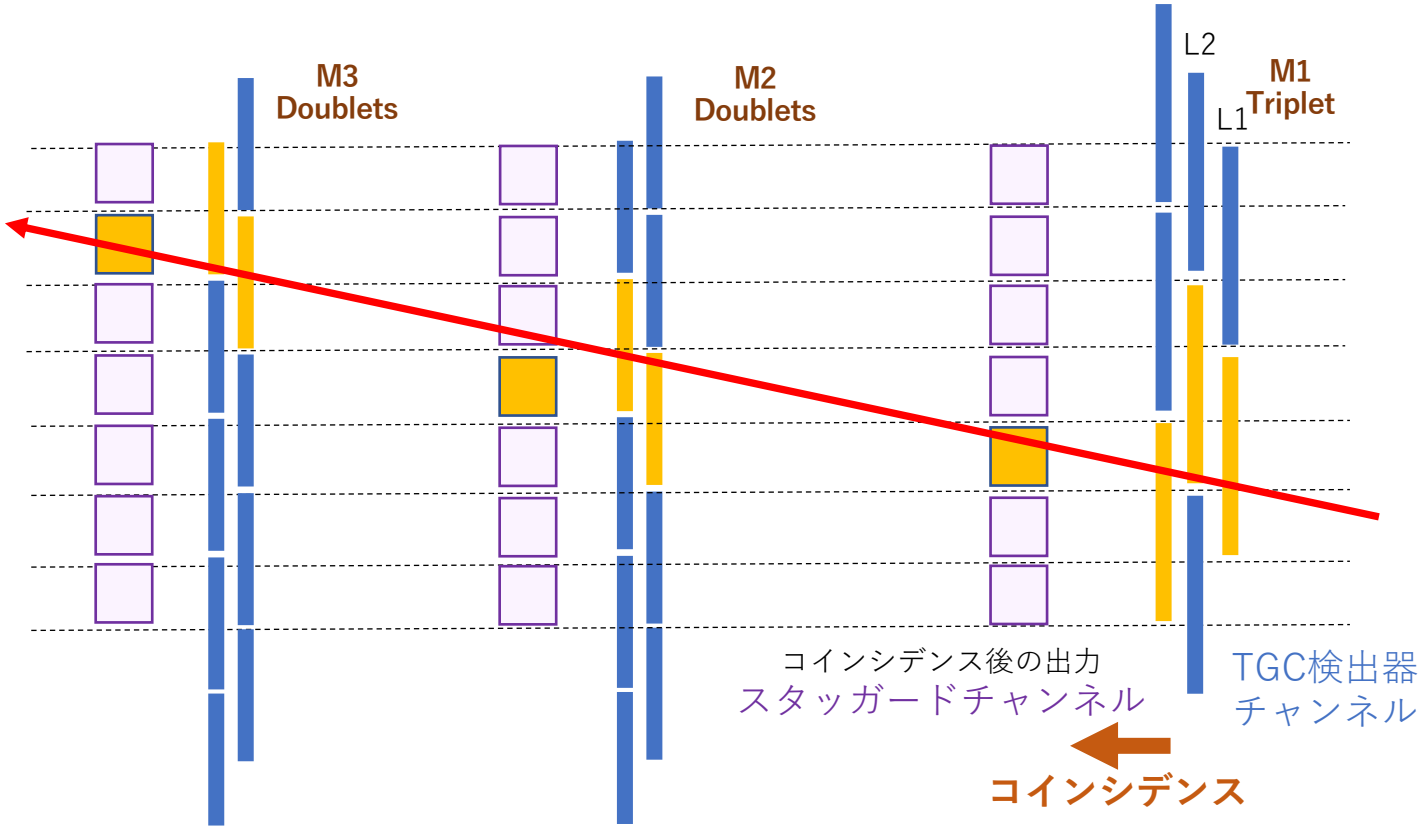


スタッガードチャンネル

- 二段階での飛跡再構成
  - I. ステーション内コインシデンス
  - II. ステーション間コインシデンス
- TGCチャンネルのスタッガリング構造
  - コインシデンスによりgranularityが向上する設計
  - トリガーで使用する位置情報が同じη/ϕのgranularityになるように検出器内のチャンネル幅が設計されている
  - コインシデンス論理はAND、OR、NOTで記述される
    - 3/3コインシデンスと2/3コインシデンスを独立＆排他的に計算する
- スタッガードチャンネルの定義
  - 同じη,ϕのチャンネルが同じ番号になるように、ステーション間で統一的なチャンネル番号の定義を行なった
    （同じチャンネル番号の組＝無限運動量飛跡）

$$3/3 = A1 \,\&\, B1 \,\&\, C2$$
$$2/3 = A1 \,\&\, B1 \,\&\, \overline{C1} \,\&\, \overline{C2} + A1 \,\&\, \overline{B1} \,\&\, \overline{B2} \,\&\, C2 + \overline{A1} \,\&\, B1 \,\&\, C2$$
$$1/3 = A1 \,\&\, \overline{B1} \,\&\, \overline{B2} \,\&\, \overline{C1} \,\&\, \overline{C2}$$

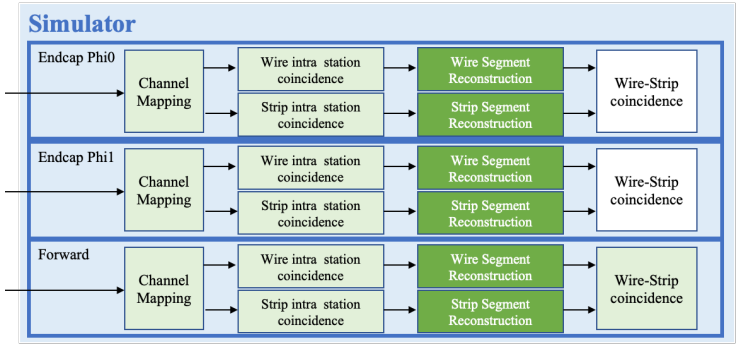




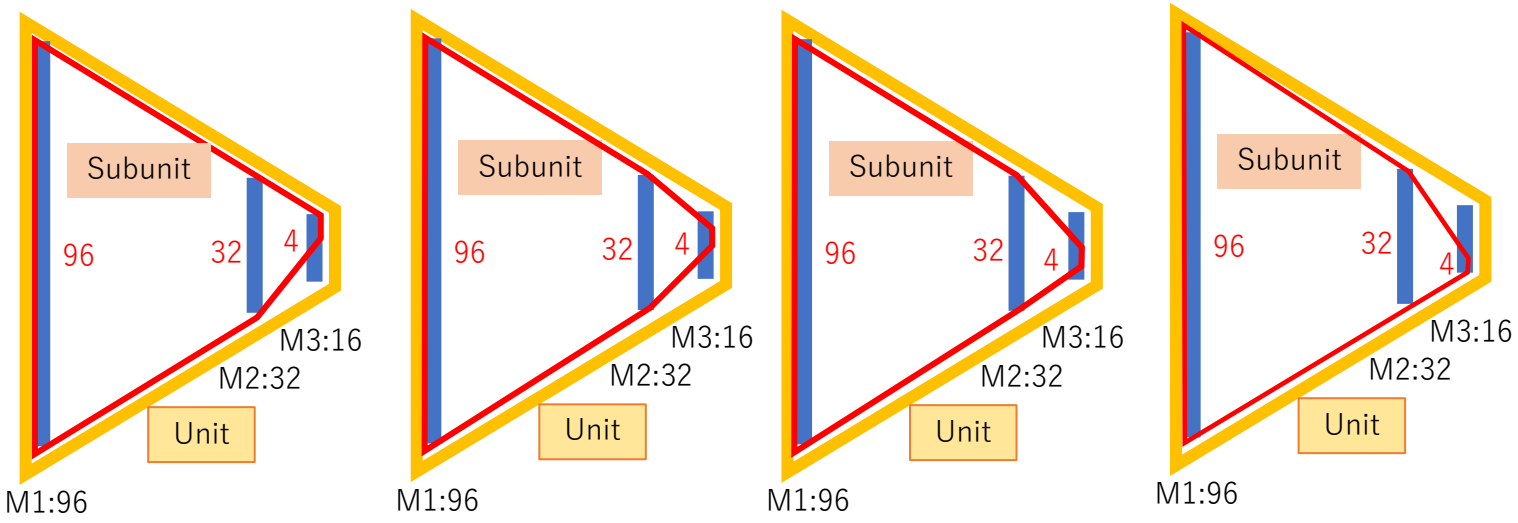
※M1で３層あるのはワイヤーのみ

Segment Reconstruction

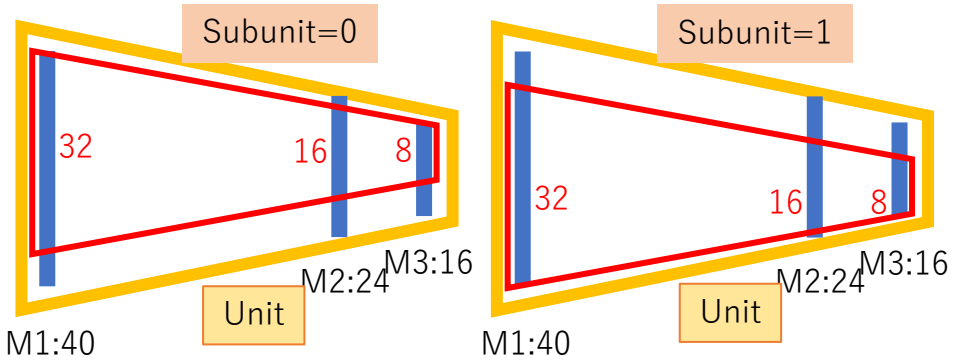
- 各ステーションの代表点の組み合わせから、無限運動量飛跡との差分$(\Delta\theta, \Delta\phi)$を得る
- Wire、Stripで独立
- 全体をUnit/subunit の構造に分割して計算を行う



Wire



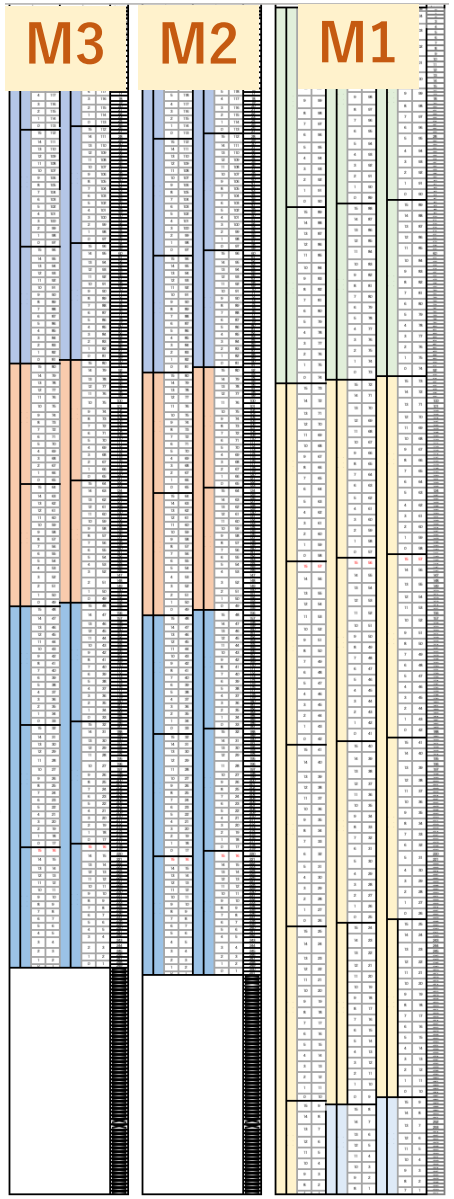
Strip



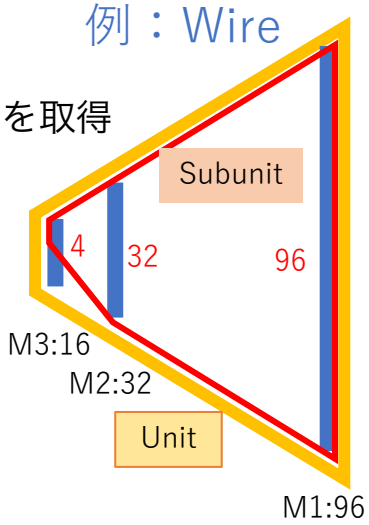
Segment Reconstruction ~ WS coincidence

- Segment Reconstruction
  - LUTにより、各ステーションの代表点の組み合わせから無限運動量飛跡との差分$(\Delta\eta, \Delta\phi)$を得る
    - トリガーは高速な計算を求められるため、セクターを複数の領域(unit/subunit)に分割し、再構成の計算を並列に行う
    - RAMによってLUTが実装される
      - subunitごとにLUTが用意される
      - LUTの入力として代表点からAddressを生成する
    - 各ステーションから「ヒットのあったレイヤー数が多く」「領域の中心に近い」代表点を取得
    - 優先順位の高いものからアドレスを生成し、LUTからデータを取得
      - Subunitという単位ごとにLUTから最大8つのデータを取得できる
    - 出力はSubunitごとに最大1データ
- Wire-Strip coincidence
  - $(\Delta\eta, \Delta\phi)$からLUTによってp_Tを得る
    - Segment Reconstructionと同様に、セクターを分割して並列計算
- C++での実装
  - シミュレータでLUTの検証をするため、LUTのパターンファイルもファームウェアと同一
    - 並列のための分割・多段の計算により、前二つのモジュールよりも非常に複雑
  - 高速化のため、ヒットのない領域などは適宜演算をスキップし、領域のセレクションも最初に行う
    - ファームウェアでは並列で演算するが、C++のシミュレータは直列的な構造のため

M1~M3ステーションについて、同じηを観測するチャンネルを横に並べた図

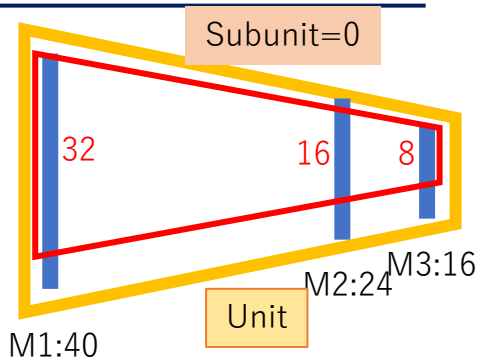


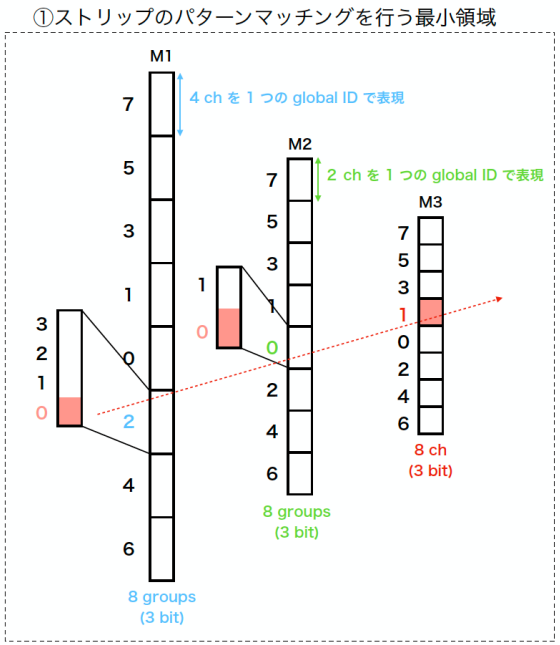
例：Wire

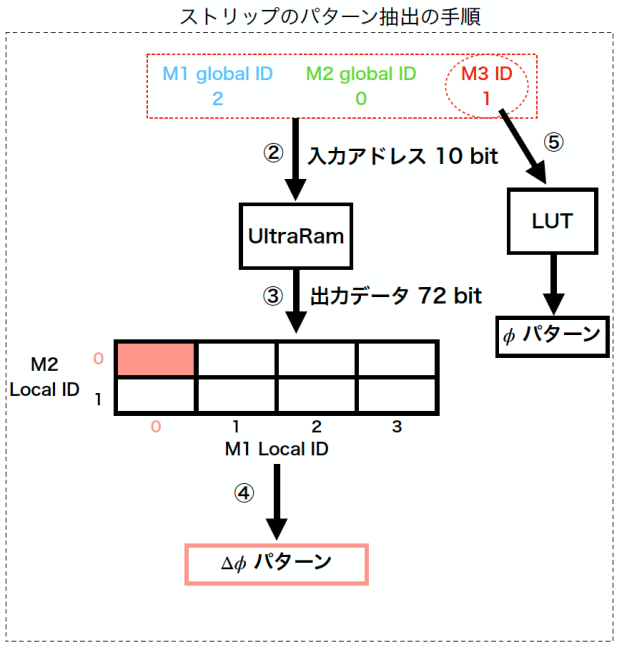


Segment Reconstruction

- LUTを使用する（ファームウェアではRAMにより実装）
  - 代表点の組み合わせがRAMのアドレスになる
    - SubunitごとにLUT
    - 例：Stripなら、
      Address = {M1 global ID (3~bit)、M2 global ID (3~bit)、M3 global ID (3~bit)}
  - 優先順位の高いものからAddressを作成
    1. ヒットのあったレイヤーの枚数
    2. Unit/subunitの中心に近い代表点
    - Strip：6 Address/subunit
    - Wire：2 Address/subunit

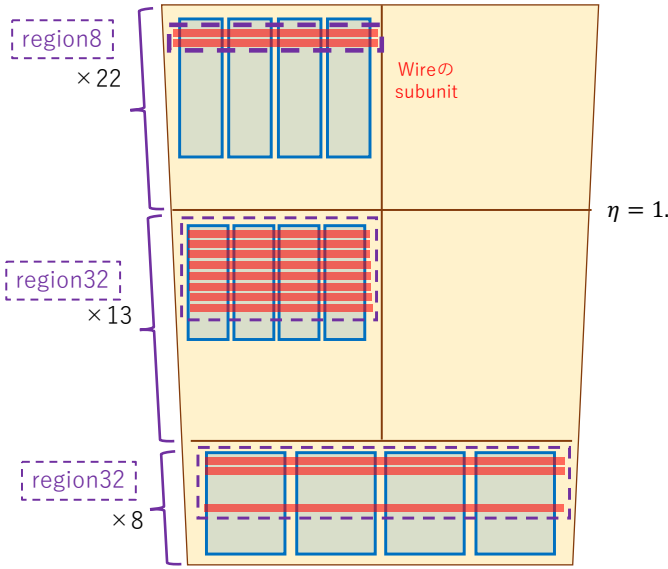


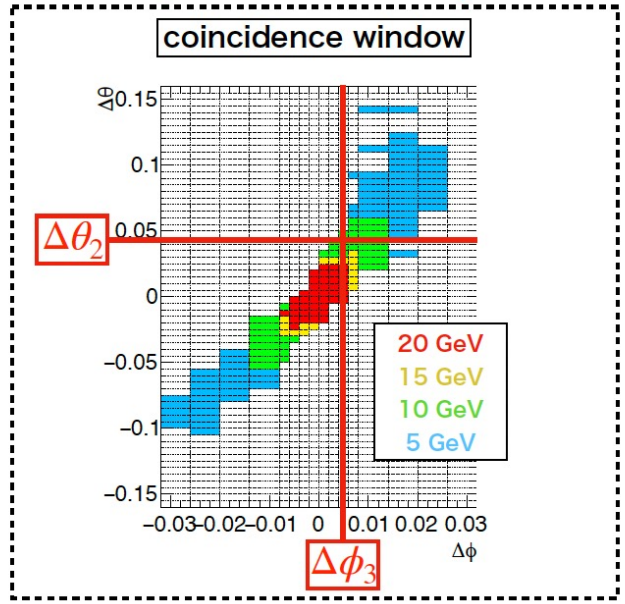




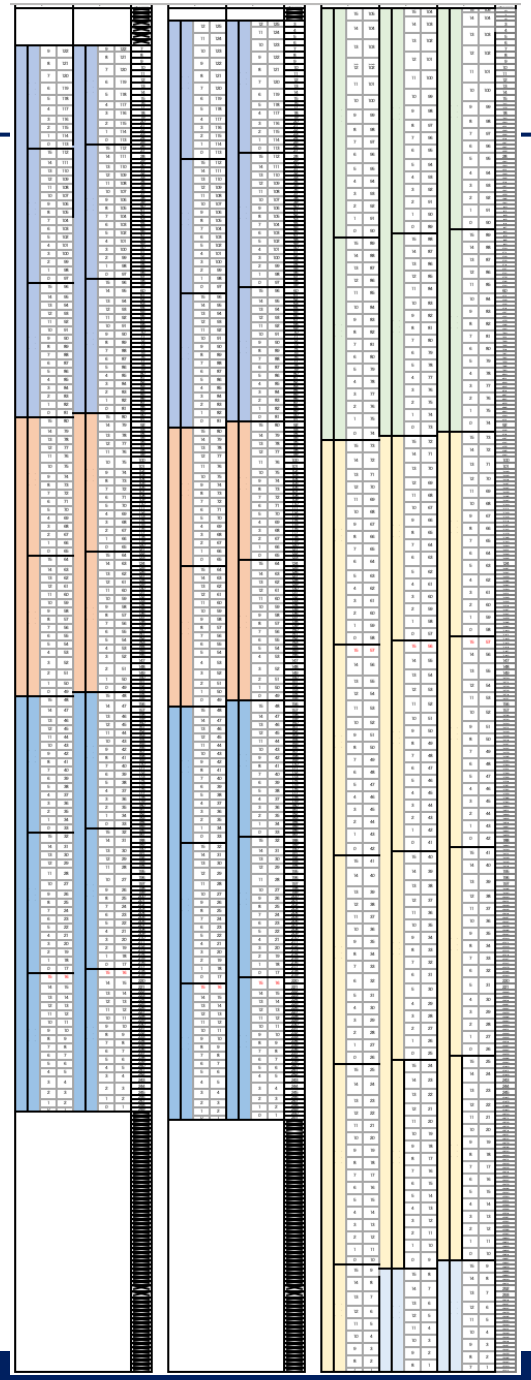
Wire-Strip Coincidence

- Segment Reconstructionで得た$(\Delta\theta, \Delta\phi)$から LUTによって$p_T$を得る
- block(最小単位): Wireのsubunit × Stripのunit
  - Wireのsubunitにつき$\Delta\theta$が最大1つ、 Stripのunitにつき$\Delta\phi$が最大1つなので、 この領域には最大1つずつの $(\Delta\theta, \Delta\phi)$が入力される
  - $(\Delta\theta, \Delta\phi)$を入力とするLUTがblockごとに用意される
- Region: 複数のblockからなる単位
  - 8つのblockで構成されるregion8と、 32のblockで構成されるregion32の2種類
    - region8: 最大1blockからの計算結果を後段に出力
    - region32: 最大4blockからの計算結果を後段に出力
  - 本研究で検証を行なったForward領域はregion32のみ

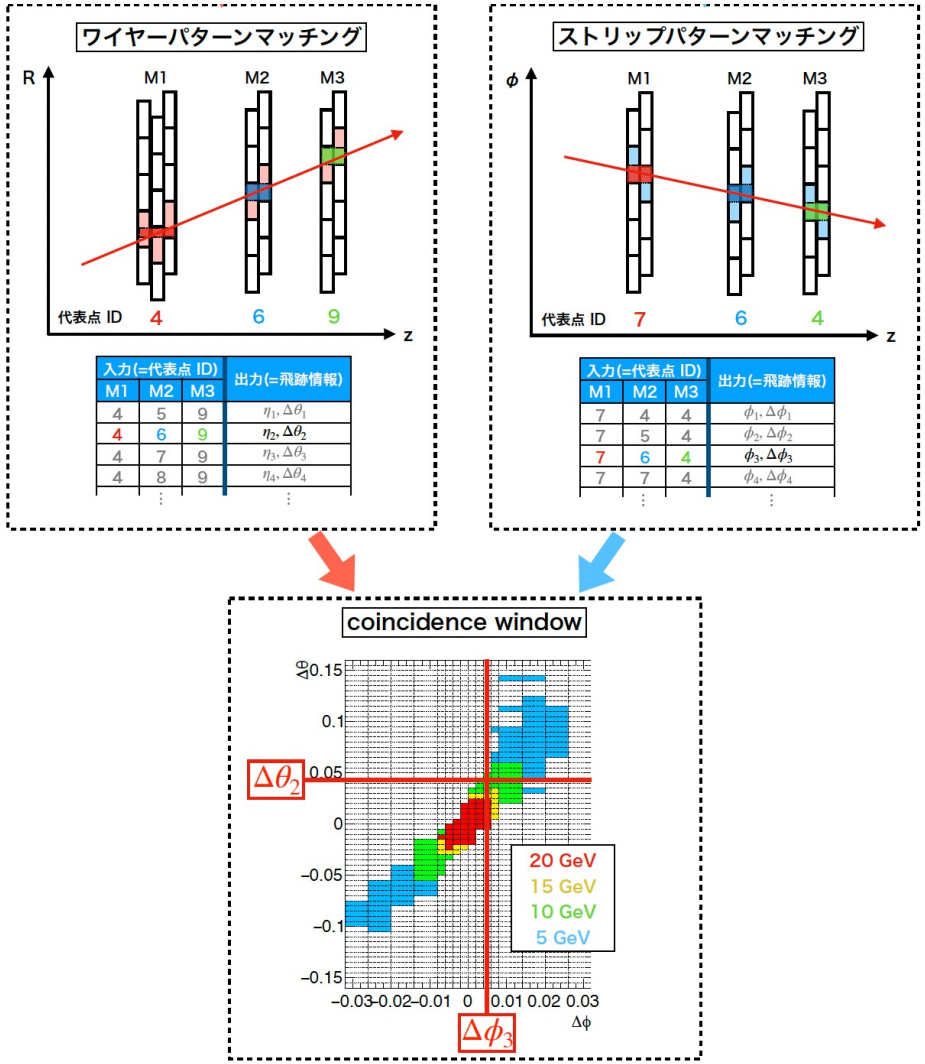




FWチャンネル

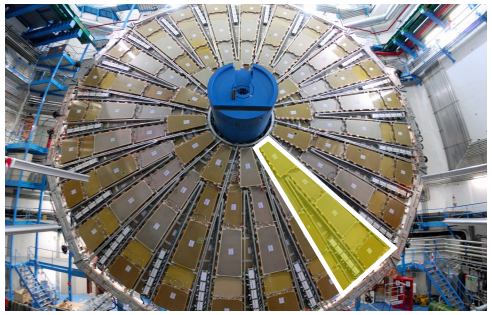


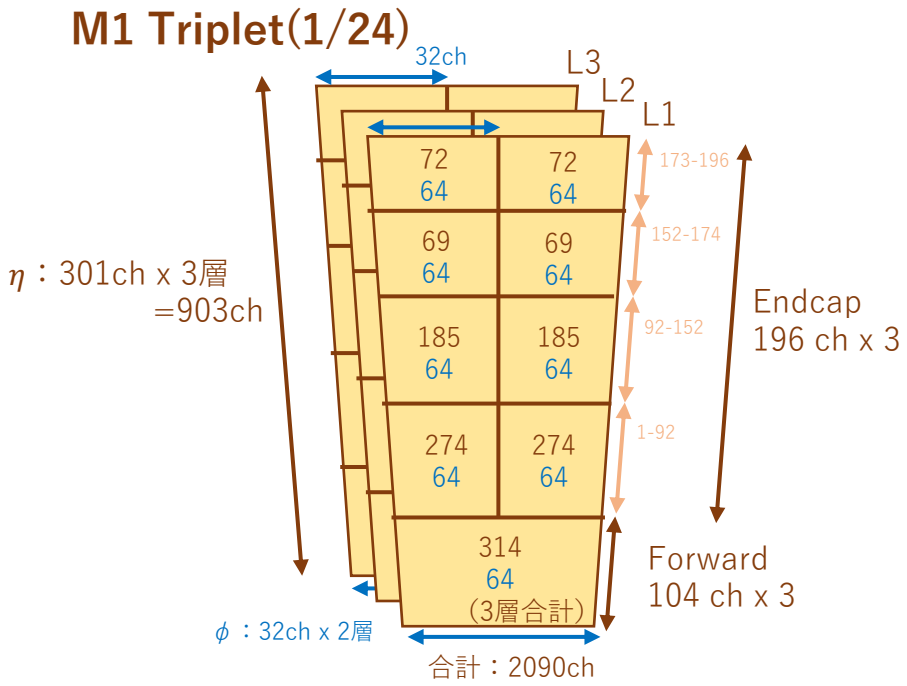
- ステーションにより η のカバー範囲が違っており、M1ステーションしか存在しない領域ではSegment Reconstructionができない

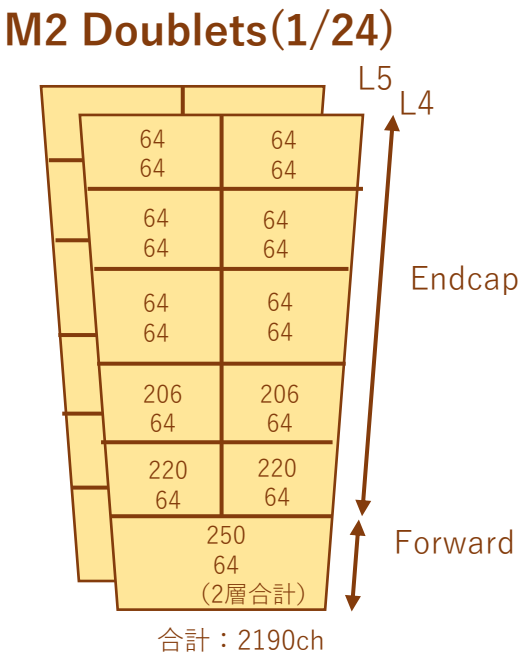

ワイヤーパターンマッチング
R
M1
M2
M3
代表点 ID 4 6 9
z
入力(=代表点 ID)
M1 M2 M3
出力(=飛跡情報)
4 5 9 $\eta_1, \Delta\theta_1$
4 6 9 $\eta_2, \Delta\theta_2$
4 7 9 $\eta_3, \Delta\theta_3$
4 8 9 $\eta_4, \Delta\theta_4$
ストリップパターンマッチング
ϕ
M1
M2
M3
代表点 ID 7 6 4
z
入力(=代表点 ID)
M1 M2 M3
出力(=飛跡情報)
7 4 4 $\phi_1, \Delta\phi_1$
7 5 4 $\phi_2, \Delta\phi_2$
7 6 4 $\phi_3, \Delta\phi_3$
7 7 4 $\phi_4, \Delta\phi_4$
coincidence window
$\Delta\theta$
0.15
0.1
0.05
0
−0.05
−0.1
−0.15
$\Delta\theta_2$
20 GeV
15 GeV
10 GeV
5 GeV
−0.03 −0.02 −0.01 0 0.01 0.02 0.03
$\Delta\phi$
$\Delta\phi_3$

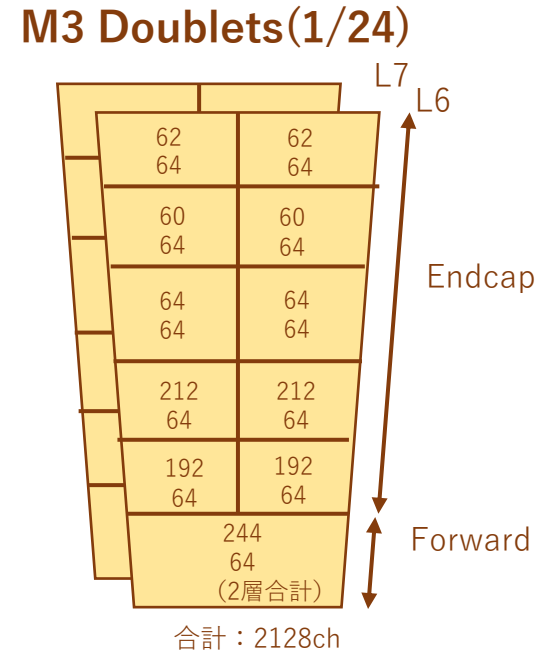
TGCを構成するチャンネル数・コンポーネント数の規模

- TGC検出器システムでは1/24構造の繰り返しで2πをカバー
- Big Wheelの1/24セクター内において以下のコンポーネントが存在している
  - 6408　チャンネル
  - 417 ASD
  - 29 PSボード
  - 58ファイバー

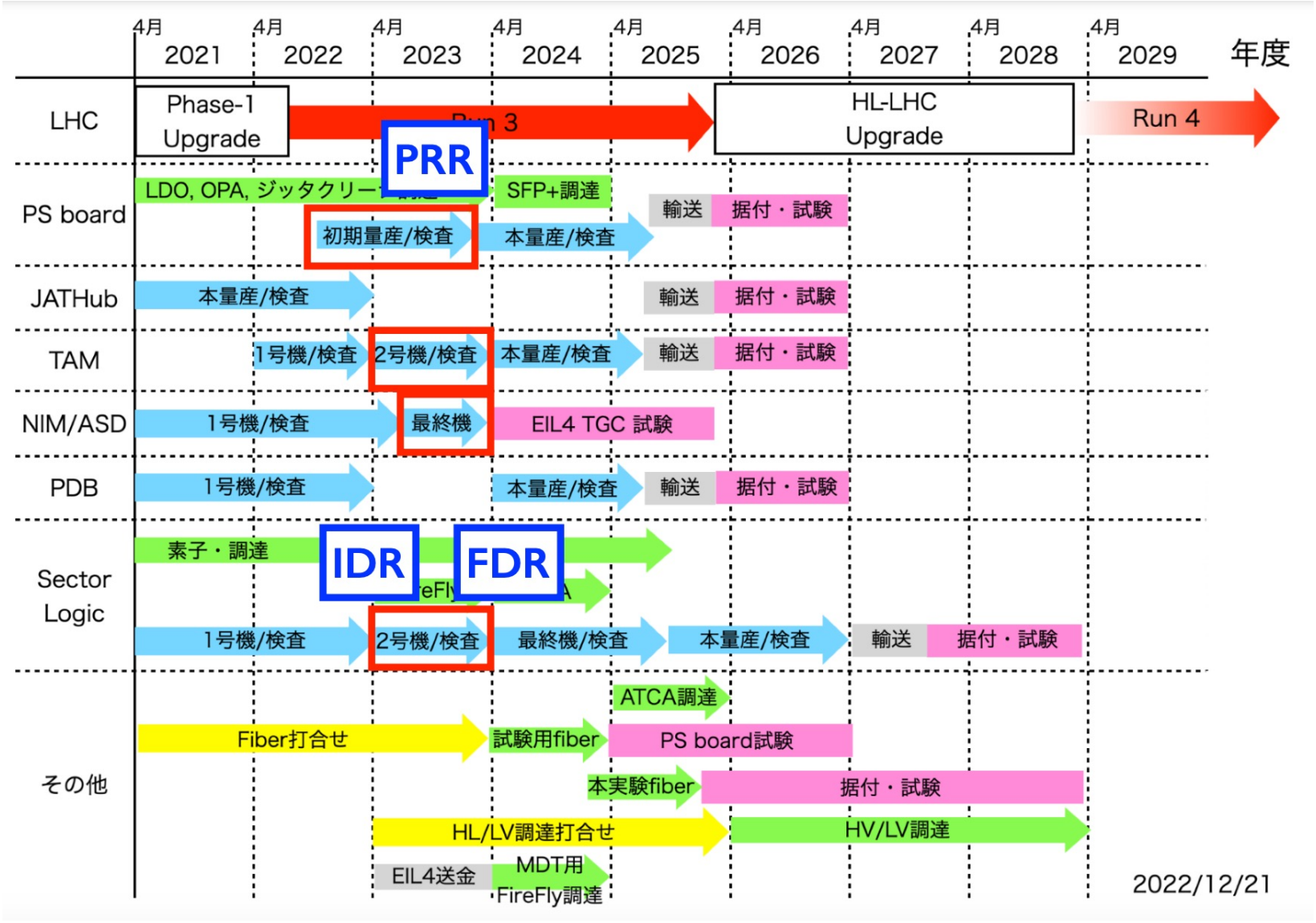


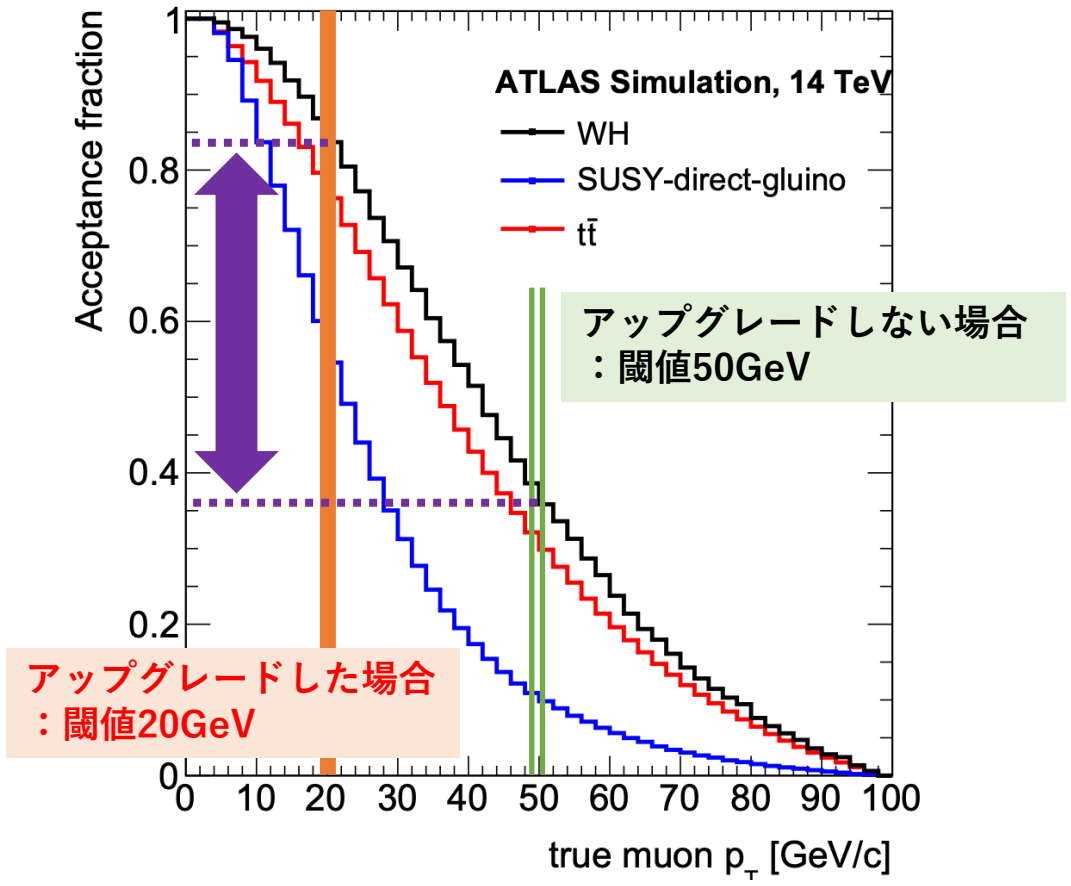




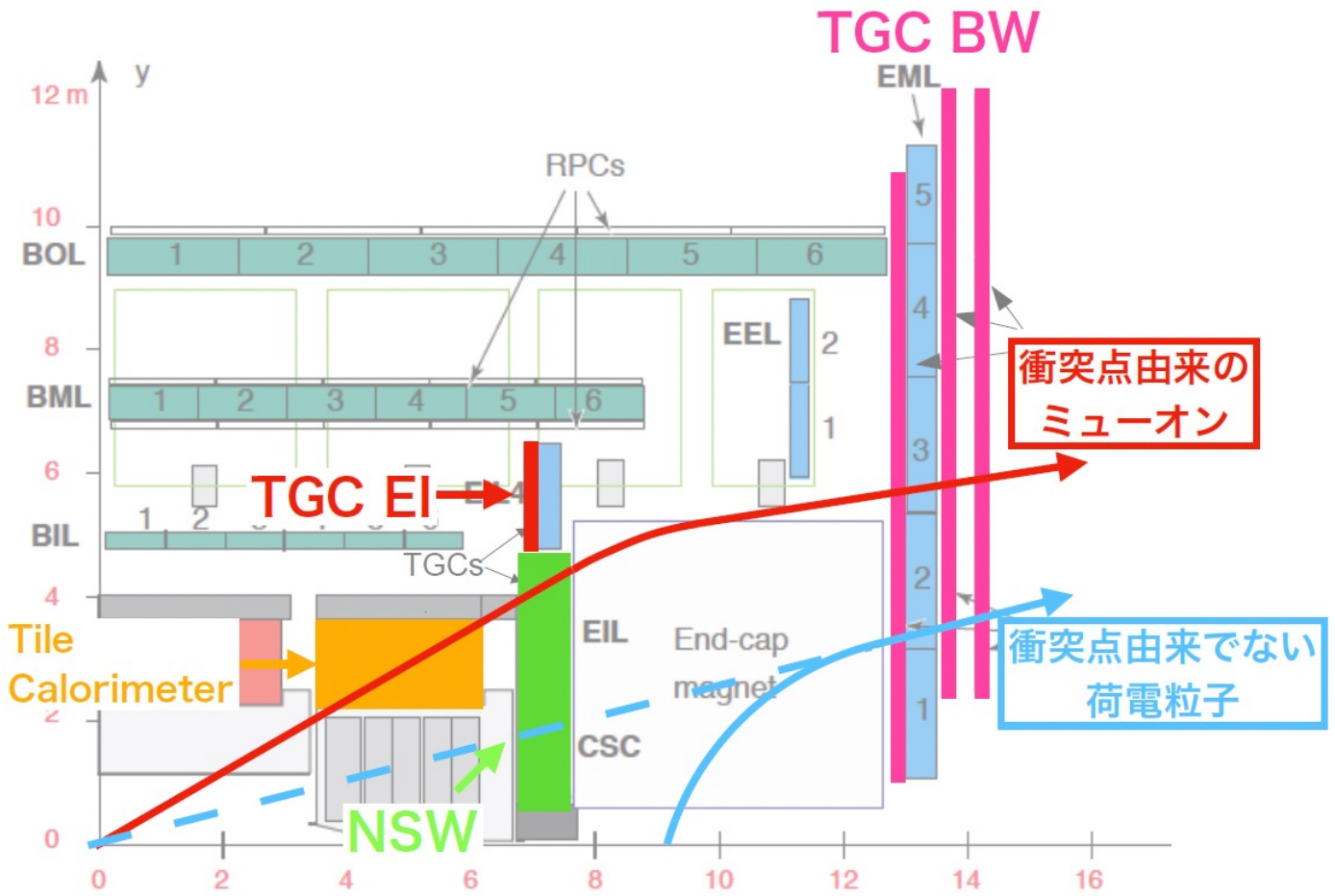


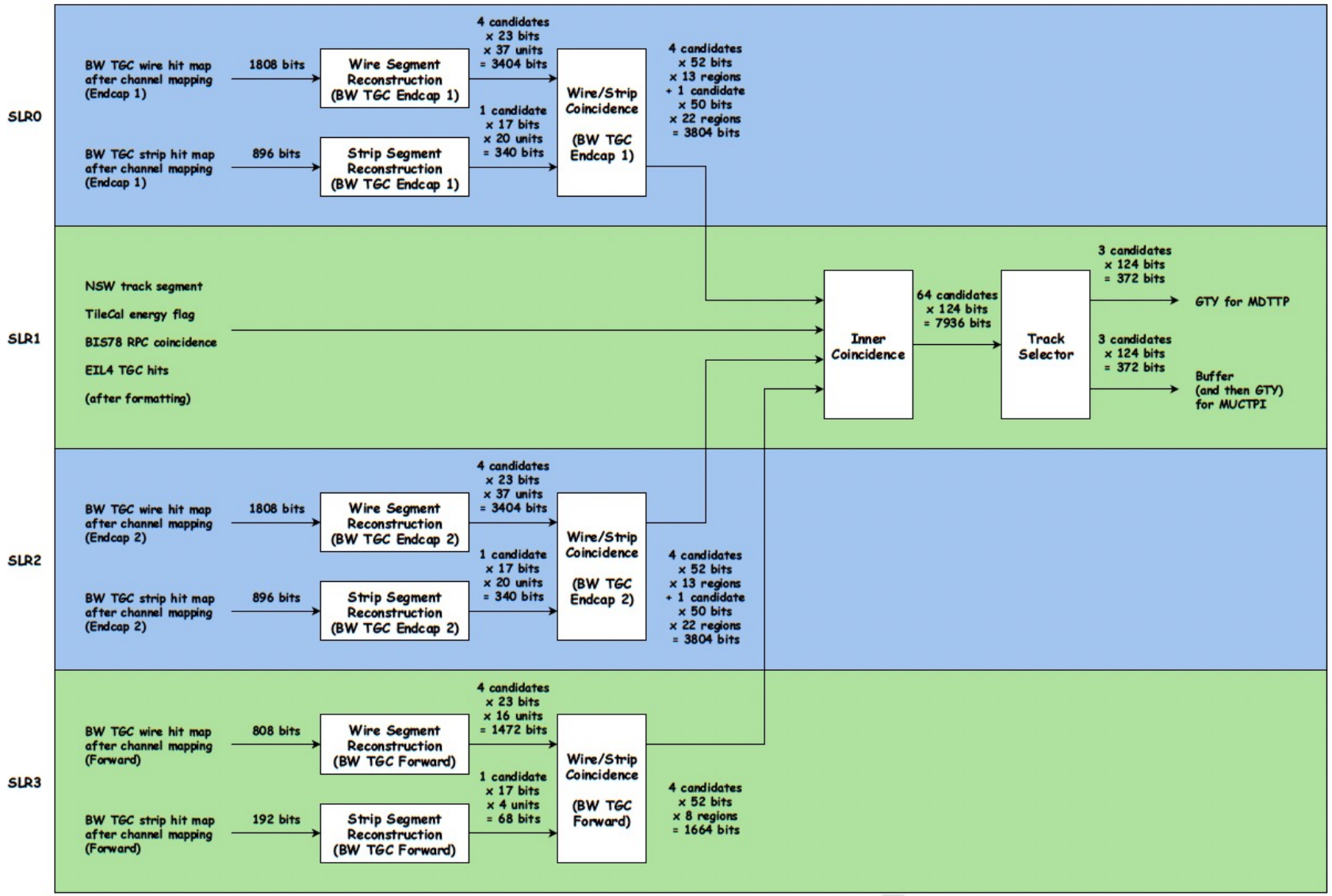
タイムスケジュール


4月 2021
4月 2022
4月 2023
4月 2024
4月 2025
4月 2026
4月 2027
4月 2028
4月 2029
年度
LHC
Phase-1 Upgrade
Run 3
HL-LHC Upgrade
Run 4
PRR
PS board
LDO, OPA, ジッタクリー
SFP+調達
輸送
据付・試験
初期量産/検査
本量産/検査
JATHub
本量産/検査
輸送
据付・試験
TAM
1号機/検査
2号機/検査
本量産/検査
輸送
据付・試験
NIM/ASD
1号機/検査
最終機
EIL4 TGC 試験
PDB
1号機/検査
本量産/検査
輸送
据付・試験
Sector Logic
素子・調達
IDR
FDR
1号機/検査
2号機/検査
最終機/検査
本量産/検査
輸送
据付・試験
その他
ATCA調達
Fiber打合せ
試験用fiber
PS board試験
本実験fiber
据付・試験
HL/LV調達打合せ
HV/LV調達
EIL4送金
MDT用 FireFly調達
2022/12/21



- $WH \rightarrow \mu\nu b\bar{b}$
  - トリガーしたい事象の一つ
  - Wボソンの崩壊で高いp_Tを持つμが生成される


TGC BW
EML
y
12 m
RPCs
10
BOL
1
2
3
4
5
6
8
BML
EEL
2
1
6
TGC EI
BIL
TGCs
4
Tile
Calorimeter
EIL
End-cap
magnet
CSC
NSW
0
0
2
4
6
8
10
12
14
16
衝突点由来の
ミューオン
衝突点由来でない
荷電粒子


SLR0
BW TGC wire hit map after channel mapping (Endcap 1)
1808 bits
Wire Segment Reconstruction (BW TGC Endcap 1)
4 candidates x 23 bits x 37 units = 3404 bits
BW TGC strip hit map after channel mapping (Endcap 1)
896 bits
Strip Segment Reconstruction (BW TGC Endcap 1)
1 candidate x 17 bits x 20 units = 340 bits
Wire/Strip Coincidence (BW TGC Endcap 1)
4 candidates x 52 bits x 13 regions + 1 candidate x 50 bits x 22 regions = 3804 bits
SLR1
NSW track segment
TileCal energy flag
BIS78 RPC coincidence
EIL4 TGC hits
(after formatting)
Inner Coincidence
64 candidates x 124 bits = 7936 bits
Track Selector
3 candidates x 124 bits = 372 bits
GTY for MDTTP
3 candidates x 124 bits = 372 bits
Buffer (and then GTY) for MUCTPI
SLR2
BW TGC wire hit map after channel mapping (Endcap 2)
1808 bits
Wire Segment Reconstruction (BW TGC Endcap 2)
4 candidates x 23 bits x 37 units = 3404 bits
BW TGC strip hit map after channel mapping (Endcap 2)
896 bits
Strip Segment Reconstruction (BW TGC Endcap 2)
1 candidate x 17 bits x 20 units = 340 bits
Wire/Strip Coincidence (BW TGC Endcap 2)
4 candidates x 52 bits x 13 regions + 1 candidate x 50 bits x 22 regions = 3804 bits
SLR3
BW TGC wire hit map after channel mapping (Forward)
808 bits
Wire Segment Reconstruction (BW TGC Forward)
4 candidates x 23 bits x 16 units = 1472 bits
BW TGC strip hit map after channel mapping (Forward)
192 bits
Strip Segment Reconstruction (BW TGC Forward)
1 candidate x 17 bits x 4 units = 68 bits
Wire/Strip Coincidence (BW TGC Forward)
4 candidates x 52 bits x 8 regions = 1664 bits

シミュレータ実装の現状

- 1枚のSLは1/24セクター（=“トリガーセクター”）を担う
- トリガーセクターは3つのサブセクターに分割され、ファームウェアはサブセクターごとに独立
  - フォワード x 1：$1.92 < |\eta| < 2.4$
  - エンドキャップ x 2： $1.05 < |\eta| < 1.92$
- 最後のセクションは図で示されるモジュールの検証を行なった話