修士学位論文

LHC-ATLAS実験Run 3の開始に向けたミューオントリガー回路系の高速読み出しと統合制御の実現

(The Realization of High-Speed Readout and Integrated Control of the Muon Trigger System for LHC-ATLAS Experiment Run 3)

東京大学大学院 理学系研究科 物理学専攻
素粒子物理国際研究センター 奥村研究室

杉崎 海斗

2021年2月28日

概要

Large Hadron Collider (LHC) は欧州原子核機構 (CERN) によって建設された世界最高エネルギーを誇る陽子陽子衝突型の円形加速器である。LHC-ATLAS 実験では、ATLAS 検出器を用いて陽子陽子衝突によって生成された粒子を観測することで、素粒子の標準模型の精密測定や標準模型を超える新物理の探索を行なっている。

陽子陽子衝突実験では衝突の全断面積が極端に大きい一方で、新物理によって予言される現象の断面積は一般にその全断面積よりも 12 桁以上も小さい。このような実験環境で限られた読み出し帯域とオフラインの計算リソースを最大限に活用するためには、重要な衝突事象のみをオンラインで選別するトリガーシステムが必要となる。LHC-ATLAS 実験では、Level-1 トリガーという初段のハードウェアトリガーを用いて 2.5 μs 以内にトリガー判定を行い、事象レートを 40 MHz から 100 kHz まで落とす。その Level-1 トリガーの一種である Level-1 ミューオンエンドキャップトリガーでは、Thin Gap Chamber (TGC) 検出器を用いて陽子陽子衝突に由来するミューオンを検出し、概算したそれらの運動量を基に事象選別を行う。

2021 年 1 月現在、LHC-ATLAS 実験では第三期運転 (Run 3) に向けてアップグレード作業を進めている。特に Level-1 ミューオンエンドキャップトリガーでは、陽子陽子衝突から生じたミューオンに由来しない偽のトリガー信号を削減するべく、New Small Wheel や RPC BIS78 という新しいミューオントリガー検出器を導入する。これらの検出器の導入に伴い、トリガー判定を担うエレクトロニクスも新たに開発された。その中核をなす 72 台の New Sector Logic (New SL) ボードは、5 種類の検出器エレクトロニクスからミューオンのヒット情報を受信し、Level-1 ミューオンエンドキャップトリガーの最終的なトリガー判定を出力する。またこのトリガーの性能をオフラインで評価するために、Software-based ROD (SROD) という新たな読み出しソフトウェアを用いることで、New SL のトリガー論理回路の入出力データを読み出す。

本研究では全 72 台の New SL を中心とする Level-1 ミューオンエンドキャップトリガーの新しいエレクトロニクスを同期制御するためのオンラインソフトウェアを開発し、システム全体の統合制御を実現した。まずは New SL を中心とした新しいエレクトロニクスやソフトウェアアプリケーションを ATLAS 検出器の制御系に組み込むために、対応するオブジェクトおよびそれらの依存関係を ATLAS のソフトウェアデータベースである Object Kernel Support (OKS) に定義した。次に、各エレクトロニクスの初期化やエレクトロニクス間の光通信リンクの確立を正しい手順で行えるように、エレクトロニクス制御アプリケーションを開発した。これらの OKS とソフトウェアアプリケーションの開発によって、Run 3 の新しい Level-1 ミューオンエンドキャップトリガーシステムが Run 2 でも使用していた TGC の検出器システムと同期して制御できるようになった。さらに新しいエレクトロニクスのモニタリング機能を実装することで、検出器システムの誤動作やその兆候を検知できるようにし、高い堅牢性を持つシステムを構築した。このような形で統合制御システムを開発することで、New SL を中心とした大規模システムを運転制御できるようにし、Run 3 に向けたコミッショニングを大幅に加速させた。

また新しいトリガーデータ読み出しソフトウェアである SROD を開発することで、100 kHz におけるデータ読み出しを実現し、検出器フロントエンドから CERN のコンピューティングセンターまでのデータ読み出しパスを確立した。はじめに、ATLAS 回路室に新たに導入された全 6 台の PC 上で、ATLAS 検出器の制御系の一部として SROD が起動するように、SROD の OKS とソフトウェアアプリケーションを開発した。さらに幾度にもわたる試験と性能改善を通して、100 kHz でのトリガーデータの読み出しを可能にするとともに、安定した動作を保証するための機能を実装した。最後に、本研究で開発した SROD ソフトウェアおよび確立した読み出しパス全体の性能評価を行うことで、これらが Run 3 の物理ランで安定動作するために十分な性能を持つことを確認した。これらの開発研究を通して、Run 3 に向けた Level-1 ミューオンエンドキャップトリガーシステムの高速読み出しと統合制御を実現した。

目次

第 1 章　序論　1

第 2 章　LHC-ATLAS 実験が目指す物理　3
- 2.1　素粒子の標準模型　3
  - 2.1.1　標準模型の概要　3
  - 2.1.2　標準模型が抱える問題点　5
- 2.2　標準模型を超える物理　6
  - 2.2.1　超対称性　6
  - 2.2.2　余剰次元　8

第 3 章　LHC-ATLAS 実験　10
- 3.1　LHC 加速器　10
- 3.2　ATLAS 検出器　12
  - 3.2.1　座標系　12
  - 3.2.2　超伝導電磁石　13
  - 3.2.3　内部飛跡検出器　14
  - 3.2.4　カロリメータ　15
  - 3.2.5　ミューオンスペクトロメータ　15
  - 3.2.6　トリガーシステム　20
  - 3.2.7　Timing, Trigger and Control (TTC) システム　21

第 4 章　Level-1 ミューオンエンドキャップトリガー　24
- 4.1　Run 2 における Level-1 ミューオンエンドキャップトリガー　24
  - 4.1.1　TGC Big Wheel のトリガーセクターと RoI　24
  - 4.1.2　TGC Big Wheel におけるトリガーの概要　24
  - 4.1.3　トリガーエレクトロニクスとそのロジック　25
  - 4.1.4　TGC の読み出しエレクトロニクス　31
  - 4.1.5　TGC のフロントエンド回路制御システム　32
  - 4.1.6　TGC の TTC システム　33
- 4.2　Level-1 ミューオンエンドキャップトリガーの Run 3 アップグレード　34
  - 4.2.1　New Small Wheel (NSW)　35
  - 4.2.2　RPC BIS78　37
  - 4.2.3　新しいトリガーエレクトロニクス　38
  - 4.2.4　トリガーデータ読み出しシステムのアップグレード　43

第 5 章　Level-1 ミューオンエンドキャップトリガーの統合制御の実現　47
- 5.1　ATLAS TDAQ ソフトウェア　47

5.1.1 OKS データベースの概要と検出器システムの階層構造 48
5.1.2 TDAQ の Run Control State と State Transition 50
5.1.3 オンラインモニタリング 51
5.1.4 Run Control Panel を用いた検出器システムの運転 54
5.2 オンラインソフトウェアによる Level-1 ミューオンエンドキャップトリガーと TGC の制御の全体像 54
5.2.1 Level-1 ミューオンエンドキャップトリガーと TGC のオンラインソフトウェアに要求される機能 55
5.2.2 制御に用いる PC と SBC 56
5.2.3 制御に必要なオンラインソフトウェアパッケージ 58
5.3 Level-1 ミューオンエンドキャップトリガーおよび TGC の OKS の開発 60
5.3.1 Run 3 の新規エレクトロニクスを制御するための OKS の開発 60
5.3.2 Segments & Resources を用いたシステムの階層構造の開発 61
5.4 Run 3 に向けた State Transition の考案とオンラインソフトウェアへの実装 63
5.5 モニタリング機能の開発 66
5.5.1 I^2C 通信の概要 66
5.5.2 TTCrx のコンフィギューレーションとモニタリング機能の開発 66
5.5.3 SFP+ 光トランシーバのモニタリング機能の開発 68
5.5.4 Grafana を用いたオンラインモニタリングの実現 70
5.6 新たに開発した統合制御システムの現状と今後 72

第 6 章 Level-1 ミューオンエンドキャップトリガーの高速読み出しシステムの開発 74
6.1 Software-based ROD (SROD) の導入 74
6.1.1 トリガーデータ読み出しパスの全体像と SROD ソフトウェアの要求性能 74
6.1.2 SROD ソフトウェアの構造 75
6.2 OKS とソフトウェアアプリケーションの相互開発による読み出しパスの開通 78
6.2.1 OKS の開発による SROD の全台制御の実現 78
6.2.2 ソフトウェアアプリケーションの開発による 100 kHz でのデータ読み出しの実現 82
6.3 Busy 信号の出力処理の実装による CTP とのハンドシェイクの実現 84
6.3.1 Busy 線の配線 85
6.3.2 OKS における busy 線の定義 85
6.3.3 SROD ソフトウェアを用いた busy 信号の出力処理の実装 86
6.3.4 SROD ソフトウェアを用いた busy 信号の出力処理の動作検証 87
6.4 トリガー読み出しデータのフォーマットの検討 88
6.4.1 SROD の入力データ 89
6.4.2 SROD の出力データ 91
6.5 L1A レートが低い場合における読み出しパスの安定性の検証 93
6.5.1 L1A レートが低い場合における動作検証試験について 93
6.5.2 L1A レートが低い場合に生じる読み出しパスの遅延の原因 94
6.5.3 L1A レートが低い場合に生じる読み出しパスの遅延時間の短縮方法の提案と実装 95
6.6 SROD の性能評価 96
6.6.1 SROD 単独の性能 98
6.6.2 読み出しパス全体の性能 103
6.7 SROD を用いた統合的なコミッショニングの現状 108
6.7.1 Milestone Week における読み出しパスのバリデーション 109

6.7.2 New SL および TTCFOB の BCR delay の調節 109
6.7.3 TGC のテストパルスを用いたトリガーパスおよびデータ読み出しパスの検証試験 110

第 7 章 結論と今後の展望 114

謝辞 115

付録 A 理論の補足 117
A.1 パウリ行列、ゲルマン行列、ガンマ行列 117
A.2 素粒子の標準模型の補足 117
A.2.1 電弱統一理論とヒッグス機構 117
A.2.2 量子色力学 121

付録 B ATLAS 検出器の補足 123
B.1 内部飛跡検出器 123
B.2 カロリメータ 124
B.3 TGC 以外のミューオンスペクトロメータ 127

付録 C OKS データベースの詳細 131
C.1 OKS のオブジェクト 131
C.2 OKS データベースの制御 132
C.3 Level-1 ミューオンエンドキャップトリガーおよび TGC の OKS の詳細 133
C.3.1 OKS データベースのファイル構造 133
C.3.2 New SL クレートを制御するために開発した OKS の詳細 134
C.3.3 TGC の Segments & Resources の全体的な吟味と再編 136

付録 D State Transition やモニタリング機能についての補足 141
D.1 TTC Restart の実装方法 141
D.2 I^2C の具体的な通信手順 141
D.3 TTCrx の I2C_ID の特定方法 142

付録 E 高速読み出しシステムの開発の補足 144
E.1 トリガーデータ読み出しフォーマットの補足：New SL の入出力データ 144
E.1.1 New SL の入力データ：各検出器エレクトロニクスから受信するトラック情報 144
E.1.2 New SL の出力データ：MUCTPI に送信するトリガー判定情報 147
E.2 L1A レートが低い場合に生じる読み出しパスの遅延の原因調査の詳細 147
E.3 L1A レートが低い場合に生じる読み出しパスの遅延時間の短縮方法の検証 148

参考文献 150

図目次

2.1 MSSM におけるゲージ結合定数の統一 8

3.1 LHC 加速器の俯瞰図 10
3.2 2018 年の各バンチ交差における陽子の平均衝突回数の分布 11
3.3 ATLAS 検出器の全体図 12
3.4 ATLAS 検出器の座標系 13
3.5 ATLAS 検出器の超伝導電磁石 14
3.6 ATLAS 検出器内における μ^- の飛跡 14
3.7 内部飛跡検出器の概要 15
3.8 ATLAS 検出器のカロリメータの断面図 16
3.9 Tile Calorimeter の層の配置 16
3.10 ミューオンスペクトロメータの各検出器の配置 17
3.11 ミューオンスペクトロメータの large sector と small sector における検出器の配置 17
3.12 TGC のチェンバーとステーションの構造 18
3.13 TGC Big Wheel の unit 18
3.14 TGC の全体写真と 1/12 セクター 19
3.15 TGC EI と FI のチェンバーの配置 19
3.16 Run 2 における ATLAS の TDAQ システム 20
3.17 HLT と後段データフローの概念図 22

4.1 TGC Big Wheel の 1/12 セクターにおけるトリガーセクターと RoI 25
4.2 TGC Big Wheel のトリガーロジックの概念図 25
4.3 Run 2 における Level-1 ミューオンエンドキャップトリガーのトリガーロジックの概要 26
4.4 ASD の概要 27
4.5 Doublet ワイヤー用の SLB ASIC の概要 28
4.6 High-p_{T} ボードの概要 29
4.7 Run 2 で用いたエンドキャップ用 SL の写真 29
4.8 TGC ROD の写真 31
4.9 TGC のフロントエンド回路制御システムの概要 32
4.10 Level-1 ミューオンエンドキャップトリガーおよび TGC の TTC システムの概要 33
4.11 Level-1 ミューオンエンドキャップトリガーにおけるフェイクトリガーの模式図 35
4.12 Run 3 における Level-1 ミューオントリガーを用いて選出したミューオン候補数の η 分布の予測 . . 36
4.13 New Small Wheel の概要 36
4.14 sTGC 検出器の構造 37
4.15 Micromegas の動作原理の模式図 38
4.16 Run 3 から設置される BIS78 ステーション 38

4.17 RPC BIS78 の 1 層の断面図 39
4.18 Run 3 の Level-1 ミューオンエンドキャップトリガーシステムの全体像 39
4.19 New Sector Logic の主な機能 40
4.20 New Sector Logic の FPGA ファームウェアの全体像 41
4.21 TGC EI/FI のチェンバーと G-Link Converter の関係 42
4.22 TTC Fanout Board の主なチップと I/O 42
4.23 Level-1 ミューオンエンドキャップトリガーのデータ読み出しシステムの Run 3 アップグレード . . 44
4.24 S-LINK 通信の概念図 45
4.25 Run 3 に向けて新たに開発された PCIe S-LINK Card 46

5.1 TDAQ ソフトウェアの動作手順 48
5.2 OKS の schema ファイルと data ファイルの一例 49
5.3 OKS を用いた検出器システムの階層構造の模式図 50
5.4 TDAQ ソフトウェアの Run Control Finite State Machine 51
5.5 Information Service の GUI 53
5.6 Error Reporting Service のメッセージの一例 53
5.7 TDAQ ソフトウェアの Run Control Panel 55
5.8 Run Control Panel から見た Segments & Resources 56
5.9 USA15 における Level-1 ミューオンエンドキャップトリガーの PC と SBC およびモジュールの配置 57
5.10 USA15 の New SL クレートの写真 57
5.11 Level-1 ミューオンエンドキャップトリガーおよび TGC のオンラインソフトウェアパッケージの一覧 59
5.12 New SL の OKS オブジェクトの定義 61
5.13 DAQSlice Partition の Segments & Resources と New SL クレートのエレクトロニクスを制御する Segment 62
5.14 Run 3 の State Transition の全体像 64
5.15 I^2C の通信方式の模式図 67
5.16 TTC Fanout Board の TTCrx のレジスタの OKS 設定値 68
5.17 DAQSlice Partition における PBEAST モニタリングの実装 70
5.18 開発した Grafana の Dashboard 71
5.19 Grafana の Query 71

6.1 Run 3 における Level-1 ミューオンエンドキャップトリガーのデータ読み出しシステムの全体像 . . 75
6.2 SROD ソフトウェアの構造 76
6.3 OKS における SROD ソフトウェアのアプリケーションの定義 79
6.4 SROD Segment の構造 80
6.5 SROD アプリケーションの動作モードの定義 81
6.6 EventBuilder の性能改善のために変更した処理 83
6.7 トリガーデータ読み出しパスの 100 kHz における長期安定性試験 84
6.8 SROD に関係する busy 線の配線 85
6.9 SROD に関係する busy 線の配線の確認試験 86
6.10 Segments & Resources における ROD Busy Module の busy 信号入力の定義 87
6.11 SROD ソフトウェアの busy 信号の出力処理の動作検証試験における Run Control Panel 88
6.12 TTC Fanout Board から SROD に送られるデータのフォーマット 89
6.13 New SL から SROD に送られるデータのフォーマット 90

6.14 SROD から ROS に送るデータのフォーマット 92
6.15 Pad word の実装方法 96
6.16 Pad word 処理の動作検証試験の結果 97
6.17 L1A レート 100 kHz における EventBuilder の全体処理時間の分布 (XOFF なし) 99
6.18 L1A レート 100 kHz における EventBuilder の各処理の所要時間の分布 (XOFF なし) 100
6.19 ROS に送信するイベントデータの大きさと EventBuilder の処理速度の関係 (XOFF なし) 102
6.20 ROS に送信するイベントデータの大きさの増加に伴う処理時間の増加 (XOFF なし) 103
6.21 EventBuilder の処理速度と L1A レートの関係 (XOFF なし) 103
6.22 L1A レート 100 kHz における EventBuilder の全体処理時間の分布 (XOFF あり) 104
6.23 L1A レート 100 kHz における EventBuilder の各処理の所要時間の分布 (XOFF あり) 105
6.24 ROS に送信するイベントデータの大きさと EventBuilder の処理速度の関係 (XOFF あり) 107
6.25 ROS に送信するイベントデータの大きさの増加に伴う処理時間の増加 (XOFF あり) 107
6.26 EventBuilder の処理速度と L1A レートの関係 (XOFF あり) 109
6.27 SROD と TGC ROD の読み出しデータの BCID の差分 110
6.28 BCR Delay 調節後の SROD と TGC ROD の読み出しデータの BCID の差分 111
6.29 TGC のテストパルスを用いた検証試験のセットアップの模式図。 112
6.30 TGC のテストパルスを用いた検証試験において SFO に記録されたトリガーデータ 112

B.1 バレル電磁カロリメータの構造 125
B.2 Tile Calorimeter のモジュールの構造 126
B.3 ハドロンカロリメータ層の interaction length 127
B.4 MDT のチューブとチェンバーの構造 128
B.5 CSC の全体図 129
B.6 CSC のアノードワイヤーとカソードストリップ 129
B.7 RPC のチェンバーの断面図 130

C.1 OKS データベースのファイル構造 133
C.2 DAQSlice Partition の Segments & Resources の全体像 136
C.3 TGC-A および TGC-C Segment の構造 137
C.4 A-side の ROD クレートと CCI クレートを制御するための Segment の構造 138
C.5 TGC Sector の ResourceSet の構造 139
C.6 TGCTTC Segment の構造 140

D.1 New SL のオンラインソフトウェアでの TTC Restart の実装 142
D.2 I^2C の通信手順 142

E.1 New SL が受信する TGC BW のトラックデータのフォーマット 145
E.2 エンドキャップ New SL が受信する TGC EI/FI のトラックデータのフォーマット 145
E.3 エンドキャップおよびフォワード New SL が受信する NSW のトラックデータのフォーマット . . . 146
E.4 エンドキャップ New SL が受信する RPC BIS78 のトラックデータのフォーマット 146
E.5 エンドキャップ New SL が受信する Tile Calorimeter のトラックデータのフォーマット 146
E.6 New SL が MUCTPI に送信するトリガー判定データのフォーマット 147
E.7 Collector アプリケーションで検知した読み出しパスの遅延時間 148

表目次

2.1 標準模型のフェルミオンとその性質 4
2.2 標準模型のボソンとその性質 4
2.3 超対称粒子の一覧 7

3.1 Run 2 と Run 3 のビームパラメータの比較 11

4.1 SROD PC の主な仕様 44
4.2 PCIe S-LINK Card の主な仕様 45

5.1 Run Control Command の概要 52
5.2 Error Reporting Service の Severity 54
5.3 New Sector Logic の OKS class の定義 61
5.4 TTCrx の代表的なレジスタ 68
5.5 TTCrx の I^2C ポインタレジスタとデータレジスタ 68
5.6 SFP+ 光トランシーバのモニタリングできるパラメータ 69

6.1 SROD Segment の名称と担当する New SL の対応表 80
6.2 SRODCollectorModule class の定義 81
6.3 New SL が SROD に送る 1 BC あたりの最大データ量 91
6.4 SROD が ROS に送るトラック情報の最大データ量 92
6.5 L1A レートが低い場合における読み出しパスの動作試験の結果 94
6.6 EventBuilder の性能評価試験における各バッファの大きさ 97
6.7 EventBuilder の各処理の平均所要時間 (XOFF なし) 101
6.8 EventBuilder の各処理の平均所要時間 (XOFF あり) 106

B.1 電磁カロリメータの各 $|\eta|$ における granularity 125

C.1 New Sector Logic の OKS class の定義 134
C.2 TTC Fanout Board の OKS class の定義 134
C.3 G-Link Converter の OKS class の定義 135
C.4 Burst Stopper の OKS class の定義 135

D.1 USA15 の TTCFOB 上の TTCrx のデフォルトの 6 ビット I2C_ID 143

E.1 SiTCP の設定を変更した場合の動作試験の結果 149

第 1 章

序論

1950 年代以降、実験と理論の相互作用によって素粒子物理学は目覚ましい発展を遂げた。素粒子がどのようにして質量を獲得するかを記述するヒッグス機構、電磁気力と弱い力を一つの理論として記述する電弱統一理論、強い力を記述する量子色力学など、数多くの理論が提唱・実証された。ここに挙げたような、素粒子を記述する上で欠かせない理論はまとめて、素粒子の標準模型 (The Standard Model, SM) と呼ばれる。標準模型は実験によって高い精度で検証されているが、まだ多くの問題を抱えている。すなわち、標準模型をより深く理解し、“標準模型を超える (Beyond the Standard Model, BSM)” 物理を発見することで、標準模型が内包する問題点を解決することが現代素粒子物理学の最重要課題である。

スイスのジュネーブ市郊外に拠点を置く欧州原子核研究機構 (CERN) では、素粒子物理のさらなる発展を実現すべく、様々な研究活動が行われている。その一環として地下 100 m に建設された Large Hadron Collider (LHC) は、世界最高エネルギーを誇る、陽子陽子衝突型の円形加速器である。2018 年まで行われていた第二期運転 (Run 2) では、その重心系エネルギーは 13 TeV に到達し、最高瞬間ルミノシティは 2.1×10^{34} cm^{-2}s^{-1} を記録した。LHC が Run 2 までで蓄積したデータ量 (積分ルミノシティ) は 190 fb^{-1} である。2022 年から始まる予定である第三期運転 (Run 3) では、重心系エネルギーを 13–14 TeV とし、より多くのデータを取得することを目指している。

ATLAS 実験では、LHC の 4 つの衝突点のうちの 1 つにある ATLAS 検出器を用い、陽子陽子衝突によって生成される粒子を捉えることで、標準模型の精密測定や新物理の探索を行なっている。LHC では陽子の集団 (バンチ, bunch) 同士の交差が 40 MHz で起こり、陽子同士の衝突頻度は 2 GHz にも達する。標準模型を超える物理などの “興味がある” 現象の生成断面積は陽子陽子衝突の全断面積に比べて 10 桁以上小さいので、限られた読み出し帯域とオフラインの計算リソースを最大限有効に活用するためには、興味がある事象をオンラインで高速選別する必要がある。ATLAS 実験では、Level-1 Trigger (Level-1 トリガー) という初段ハードウェアトリガーおよび High Level Trigger (HLT) という後段ソフトウェアトリガーから構成される大規模なトリガーシステムを用いて、この高速選別を実現している。特に Level-1 トリガーでは、2.5 μs 以内にトリガー判定を行い、事象レートを 100 kHz まで落とすことが要求される。また、Level-1 トリガーは対象とするオブジェクト別に分類される複数のトリガーから構成される。その 1 つである Level-1 ミューオントリガーは、陽子陽子衝突から飛来するミューオンの横運動量 (transverse momentum, p_{T})[*1]に閾値を設けることで事象選別を行うトリガーであり、エンドキャップ部とバレル部に二分される[*2]。本研究では、Level-1 ミューオンエンドキャップトリガーに焦点を置く。

Level-1 ミューオンエンドキャップトリガーは、Thin Gap Chamber (TGC) というミューオン検出器のヒット情報を用いてトリガー判定を行う。TGC ではミューオンが残すヒット情報から磁場中の曲がり具合を見積もることでその p_{T} を概算し、その p_{T} に基づいてトリガー判定を行う。しかしトリガー判定の主役を担っている TGC Big Wheel (BW) は磁場領域外に位置するため、Run 2 ではビームパイプなどから来る荷電ハドロンによって、予想されていた以上に偽のトリガー信号が出力されたことが課題となった。Run 3 ではそれらの偽のトリガー信号を削減するべく、New Small Wheel や RPC BIS78 などの新しいミューオン検出器が磁場領域内に導入される予定であ

[*1] 横運動量とは粒子が持つ運動量のビーム軸に垂直な成分のことである。

[*2] ATLAS 検出器は円筒型であり、円筒の側面部分をバレル部、底面部分をエンドキャップ部と呼ぶ。詳細は 3.2 節で述べる。

る。それゆえ、これらの新しい検出器および Run 2 で使っていた検出器から来る信号を基にトリガー判定を行う、New Sector Logic (New SL) という、新たなバックエンドボードが開発された。これに付随して、New SL に LHC のクロック信号などを配布する TTC Fanout Board というハードウェアや、オフラインで新しいトリガーロジックの性能評価を行うために SROD というトリガーデータ読み出しソフトウェアが開発された。

この新しい Level-1 ミューオンエンドキャップトリガーの安定運転を Run 3 のデータ収集の開始とともに実現するためには、New SL を中心とした新たなトリガー回路およびトリガーデータ読み出しシステムを同期制御する機構が必須になる。具体的には、まず新たに導入するエレクトロニクスやソフトウェアを ATLAS 検出器全体の制御系に組み込み、それらの依存関係 (階層構造) を明確に定義することで、Level-1 ミューオンエンドキャップトリガーシステムの系統的な制御を実現する必要がある。またサブシステムの枠を超えて、エレクトロニクスの初期化やエレクトロニクス間の光通信リンクの確立を正しい手順で行うために、システムのすべてのエレクトロニクスの同期を可能にする運転状態の遷移を実装しなければならない。加えて、検出器エレクトロニクスの誤動作やその兆候を検知することで堅牢性に長けたシステムを構築するために、新規エレクトロニクスのモニタリングを行う必要がある。

本研究は Run 3 に向けてこの新しい Level-1 ミューオンエンドキャップトリガーシステムの同期制御およびその運転を実現するための第一歩である。オンラインソフトウェアを中心とした統合制御システムの開発によって、New SL を含む新規エレクトロニクスが Run 2 まで使われていたエレクトロニクスとの同期制御が達成される。またトリガーデータ読み出しソフトウェアの開発およびその度重なるストレステストによって、Level-1 ミューオンエンドキャップトリガーのトリガーデータ読み出しパスを確立する。2022 年に Run 3 を控えた今、物理データを収集するための基礎環境を構築するという観点から、本研究の重要度は非常に高い。本論文ではこれらの開発研究について詳しく述べる。

本論文の構成は以下の通りである。第 2 章では、素粒子の標準模型やそれを超える新物理など、LHC-ATLAS 実験が目指す物理についてまとめる。第 3 章では、LHC 加速器や ATLAS 検出器の概要を述べる。第 4 章では、現行の Level-1 ミューオンエンドキャップトリガーシステムについて、および Run 3 に向けたそのアップグレード計画について詳しく説明する。第 5 章および第 6 章では、Run 3 に向けて行なった Level-1 ミューオンエンドキャップトリガーの開発研究について議論する。第 5 章は、Level-1 ミューオンエンドキャップトリガーおよび TGC のエレクトロニクスの統合制御を実現するためのオンラインソフトウェア開発についてである。第 6 章は、新しいトリガーデータ読み出しソフトウェアである SROD の開発を中心とした、Level-1 ミューオンエンドキャップトリガーのデータ読み出しパスの確立についてである。最後に第 7 章で本研究の結論と今後の展望についてまとめる。

第 2 章

LHC-ATLAS 実験が目指す物理

本章では、エネルギーフロンティア実験である LHC-ATLAS 実験が目指す物理の理論背景についてまとめる。2.1 節では現代素粒子物理の基盤となっている素粒子の標準模型について述べる。2.2 節では LHC-ATLAS 実験でも盛んに探索解析が行われている、超対称性理論と余剰次元理論の 2 つの標準模型を超える理論について簡単に説明する。なお本章では $c = \hbar = 1$ の自然単位系を用い、Minkowski 計量は $\eta_{\mu\nu} = \mathrm{diag}(1, -1, -1, -1)$ を用いる[*1]。

2.1 素粒子の標準模型

私たちが暮らすエネルギー領域では、素粒子の間に 4 種類の相互作用がはたらくことが知られている。それらは強い順に、強い相互作用、電磁相互作用、弱い相互作用、および重力相互作用である。2021 年 1 月現在、重力を除く 3 つの相互作用は理論的に定式化されている。端的に述べると、現代素粒子物理の基盤となっている素粒子の標準模型 (The Standard Model, SM) はこれらの 3 つの相互作用によってどのように素粒子が振る舞うかを記述する理論体系である。本節では標準模型についてまとめたのち、それが内包する問題点について述べる。なお本節の議論は量子補正による効果は取り扱わず、あくまでツリーレベルであることをここで断っておく。

本題に入る前に、SM の相互作用とゲージ群の関係について述べておく。SM の相互作用を理論的に記述するには、粒子場のラグランジアン (Lagrangian) にゲージ対称性 (局所対称性) を課さなければならないことが一般的に知られている。対称性が存在するということはその背景に群が関わっていることを意味する。すなわち、SM の相互作用には必ずそれに対応するゲージ群が存在することになる。

2.1.1 標準模型の概要

電磁相互作用と弱い相互作用は、ヒッグス機構を導入した電弱統一理論という $SU(2)_L \times U(1)_Y$ ゲージ理論を用いて統一的に説明される。また、強い相互作用は量子色力学という $SU(3)_c$ ゲージ理論によって表される。すなわちこれらの 3 つの相互作用を記述する素粒子の標準模型は、$SU(3)_c \times SU(2)_L \times U(1)_Y$ ゲージ理論である。この小節では付録 A.2 に記す電弱統一理論と量子色力学の議論に基づき、標準模型の概要を説明する。

まずは表 2.1 に標準模型のフェルミオンとその性質をまとめる[*2]。標準模型のフェルミオンは強い相互作用をするクォークと、強い相互作用をしないレプトンに大きく分けられる。これらはともに 3 つの世代から構成される。今後の議論のために世代数 $i = 1, 2, 3$ を用いた表記を導入する。アップ、チャーム、トップはまとめてアップタイプクォーク $u^i = u, c, t$ と表記し、ダウン、ストレンジ、ボトムはまとめてダウンタイプクォーク $d^i = d, s, b$ と表す。またレプトンについても同様に、3 つのニュートリノは ν^i として表し、電子、ミューオン、タウは荷電レプトン e^i

[*1] $\mu, \nu = 0, 1, 2, 3$ は時空成分を表す添字である。

[*2] クォークは閉じ込め (confinement) の性質によって質量の測定が難しく、その測定基準は複数存在する。表 2.1 に示す値の基準をここに明記しておく。u, d, s の質量は $\overline{\mathrm{MS}}$ scheme などにおいて $\mu \approx 2$ GeV とした場合の見積もり値である。c, b の質量は $\overline{\mathrm{MS}}$ scheme における running mass であり、t の質量は直接測定によって得られている値である。

としてまとめて表記する。さらに標準模型の $SU(2)_L$ ゲージ群は上記のフェルミオンの左巻き成分にしか作用しない。すなわちフェルミオンの左巻き成分は $SU(2)_L$ 二重項を作り、フェルミオンの右巻き成分は $SU(2)_L$ 一重項をなす。最終的に標準模型のすべてのフェルミオンは、

$$Q_L^i = \begin{pmatrix} u_L^i \\ d_L^i \end{pmatrix}, L_L^i = \begin{pmatrix} \nu_L^i \\ e_L^i \end{pmatrix}, u_R^i, d_R^i, e_R^i \tag{2.1}$$

とまとめて表記できる[*3]。さらにクォークは $SU(3)_c$ 三重項を構成し、赤緑青のいずれかのカラーを持つが、以下の議論ではカラーはあらわに表記しない。

表 2.1　標準模型のフェルミオンとその性質 [1]。ニュートリノの質量は標準模型では 0 であると仮定されているものの実際には 0 でないことが知られているため、ここではその質量の上限値を記した[*4]。

		名称	表記	質量	電荷	スピン
クォーク	第 1 世代	アップ	u	$2.16\ ^{+0.49}_{-0.26}$ MeV	$+2/3$	$1/2$
		ダウン	d	$4.67\ ^{+0.48}_{-0.17}$ MeV	$-1/3$	$1/2$
	第 2 世代	チャーム	c	1.27 ± 0.02 GeV	$+2/3$	$1/2$
		ストレンジ	s	$93\ ^{+11}_{-5}$ MeV	$-1/3$	$1/2$
	第 3 世代	トップ	t	172.76 ± 0.30 GeV	$+2/3$	$1/2$
		ボトム	b	$4.18\ ^{+0.03}_{-0.02}$ GeV	$-1/3$	$1/2$
レプトン	第 1 世代	電子ニュートリノ	ν_e	< 1.1 eV	0	$1/2$
		電子	e	$0.5109989461 \pm 0.0000000031$ MeV	-1	$1/2$
	第 2 世代	ミューニュートリノ	ν_μ	< 0.19 eV	0	$1/2$
		ミューオン	μ	$105.6583745 \pm 0.0000024$ MeV	-1	$1/2$
	第 3 世代	タウニュートリノ	ν_τ	< 18.2 eV	0	$1/2$
		タウ	τ	1776.86 ± 0.12 MeV	-1	$1/2$

次に、表 2.2 に標準模型のボソンとその性質をまとめる。ゲージボソンには強い相互作用を媒介するグルーオン、弱い相互作用を媒介する W ボソンと Z ボソン (まとめて弱ボソンともいう)、そして電磁相互作用を媒介する光子がある。弱ボソンと光子の関係は付録 A.2.1 で補足する。またヒッグスボソンは、ヒッグス二重項として導入される 4 つの自由度のうち、$SU(2)_L \times U(1)_Y$ ゲージ対称性の自発的破れの後に残る唯一の自由度であることも付録 A.2.1 にて説明する。

表 2.2　標準模型のボソンとその性質 [1]。

	名称	表記	質量	電荷	スピン
ベクトルボソン	グルーオン	g	0	0	1
	W ボソン	$W^\pm$	80.379 ± 0.012 GeV	± 1	1
	Z ボソン	Z^0	91.1876 ± 0.0021 GeV	0	1
	光子	γ	0	0	1
スカラーボソン	ヒッグス	h	125.10 ± 0.14 GeV	0	0

さて、クォークの $SU(2)_L$ 二重項、クォークの $SU(2)_L$ 一重項、レプトンの $SU(2)_L$ 二重項、レプトンの $SU(2)_L$

*3 2021 年 1 月現在、右巻きニュートリノの存在は確認されておらず、標準模型にも含まれない。
*4 ここで示す ν_e の質量の上限値は、正確には $\bar{\nu}_e$ の上限値である。

一重項、そしてヒッグス二重項に作用する共変微分をそれぞれ $D_\mu^{Q_L}, D_\mu^{q_R}, D_\mu^{L_L}, D_\mu^{l_R}, D_\mu^H$ とすると、それらは

$$\begin{aligned} D_\mu^{Q_L} &= \partial_\mu + ig_3 G_\mu^\alpha \frac{\lambda^\alpha}{2} + ig_2 W_\mu^a \frac{\sigma^a}{2} + ig_Y B_\mu \frac{Y}{2} \\ D_\mu^{q_R} &= \partial_\mu + ig_3 G_\mu^\alpha \frac{\lambda^\alpha}{2} + ig_Y B_\mu \frac{Y}{2} \\ D_\mu^{L_L} = D_\mu^H &= \partial_\mu + ig_2 W_\mu^a \frac{\sigma^a}{2} + ig_Y B_\mu \frac{Y}{2} \\ D_\mu^{l_R} &= \partial_\mu + ig_Y B_\mu \frac{Y}{2} \end{aligned} \tag{2.2}$$

と書ける。ただし、g_3, g_2, g_Y はそれぞれ $SU(3)_c, SU(2)_L, U(1)_Y$ のゲージ結合定数である。G_μ^α $(\alpha = 1, ..., 8)$ はグルーオンに対応する $SU(3)_c$ ゲージ場、W_μ^a $(a = 1, 2, 3)$ は $SU(2)_L$ ゲージ場、そして B_μ は $U(1)_Y$ ゲージ場である。また λ^α と σ^a は付録 A.1 に示すゲルマン行列とパウリ行列で、ハイパーチャージ Y は $U(1)_Y$ ゲージ場についてのチャージである。以降では簡単のため、共変微分はすべて D_μ と略記する。

ここからは上記の表記を用いて、標準模型のラグランジアンについてまとめる。フェルミオン場の運動項は、式 (A.3) に示すディラック行列 γ^μ を用いて、

$$\mathcal{L}_F = \overline{Q_L^i} i\gamma^\mu D_\mu Q_L^i + \overline{u_R^i} i\gamma^\mu D_\mu u_R^i + \overline{d_R^i} i\gamma^\mu D_\mu d_R^i + \overline{L_L^i} i\gamma^\mu D_\mu L_L^i + \overline{e_R^i} i\gamma^\mu D_\mu e_R^i \tag{2.3}$$

と書ける。ただし第一項から第三項までのクォーク項ではカラーの寄与も考慮する。ゲージ場の運動項は、式 (A.11) および式 (A.37) に定義を示す $G_{\mu\nu}^\alpha, W_{\mu\nu}^a, B_{\mu\nu}$ を用いて、

$$\mathcal{L}_G = -\frac{1}{4} G_{\mu\nu}^\alpha G^{\alpha\mu\nu} - \frac{1}{4} W_{\mu\nu}^a W^{a\mu\nu} - \frac{1}{4} B_{\mu\nu} B^{\mu\nu} \tag{2.4}$$

となる。ヒッグスに関する項はヒッグス二重項 Φ を用いて、

$$\mathcal{L}_H = (D_\mu \Phi)^\dagger (D^\mu \Phi) - V(\Phi) \tag{2.5}$$

と書ける。これは式 (A.15) に示す電弱統一理論の場合と同じものである。標準模型における湯川相互作用項は、式 (A.32) に示す電弱統一理論の湯川相互作用項の第二項と第三項のクォーク項にカラーの寄与も取り入れた上で同様に、

$$\mathcal{L}_Y = -y_L^{ij} \left\{ \overline{L_L^{0i}} \Phi e_R^{0j} + \overline{e_R^{0i}} \Phi^\dagger L_L^{0j} \right\} - y_d^{ij} \left\{ \overline{Q_L^{0i}} \Phi d_R^{0j} + \overline{d_R^{0i}} \Phi^\dagger Q_L^{0j} \right\} - y_u^{ij} \left\{ \overline{Q_L^{0i}} \widetilde{\Phi} u_R^{0j} + \overline{u_R^{0i}} \widetilde{\Phi}^\dagger Q_L^{0j} \right\} \tag{2.6}$$

と書ける。すなわち標準模型のラグランジアン $\mathcal{L}_{SM}$ は、式 (2.3)–(2.6) をすべて足して、

$$\mathcal{L}_{SM} = \mathcal{L}_F + \mathcal{L}_G + \mathcal{L}_H + \mathcal{L}_Y \tag{2.7}$$

である。

2.1.2 標準模型が抱える問題点

標準模型は高い精度で検証されているものの、未だに多くの問題点を内包する。ここではそれらの問題のうち、代表的なものについて簡単にまとめる。

暗黒物質

我々が理解している通常の物質だけでは、銀河の回転速度分布などの天文学的な観測事実を説明できないことが知られている。このため宇宙には未だに観測できていない、質量を持つ正体不明の物質があるとされている。これらの物質は “暗黒物質 (dark matter)” と呼ばれる。現在の観測結果によると、宇宙のすべての暗黒物質が持つエネルギーは通常の物質の 5 倍程度とされる [2]。暗黒物質の候補として様々なものが挙げられているが、その最有力候

補の 1 つは未知の素粒子である。しかし暗黒物質を未知の素粒子であると仮定した場合、それは質量を持ち、通常の物質とはほとんど相互作用しないものでなければならない。素粒子の標準模型はそのような素粒子を持たない[*5]。よって素粒子的な暗黒物質を説明するためには、標準模型を拡張しなければならない。

電荷の量子化と大統一理論

標準模型は 1 つの $SU(3)_c \times SU(2)_L \times U(1)_Y$ ゲージ理論ではあるが、3 つのゲージ群は直積として表されている。つまり 3 つのゲージ群は別々の結合定数を持ち、それぞれ完全に独立したものである。このことに加え、電荷の値に寄与する $U(1)_Y$ のハイパーチャージ Y の固有値は任意である。しかしながら表 2.1 に示したように、ダウンクォークの電荷 $Q(d)$ と荷電レプトンの電荷 $Q(l)$ の間には $|Q(l)| = 3|Q(d)|$ という関係が成り立つ。標準模型の枠組では Y の固有値は任意なので、これらの電荷の大きさの間に整数倍の関係が成り立つ原理的な理由はない。これを解決するために、$SU(3)_c \times SU(2)_L \times U(1)_Y$ はより大きい 1 つのゲージ群から生じたと考える、大統一理論 (Grand Unified Theory, GUT) を導入することが必要とされている。大統一理論を導入することで、$O(10^{16}$ GeV) の高エネルギー領域において強い相互作用と電弱相互作用のゲージ結合定数が合致し、3 つの力が統一されると考えられている。すなわち標準模型は、大統一理論の低エネルギー有効理論であるという考え方が有力視されている。

ヒッグスの質量と階層性問題

2012 年に発見されたヒッグスボソンの質量は 125 GeV であることがわかっている。しかし前述したように標準模型がとある大統一理論の低エネルギー有効理論であると考えた場合、125 GeV という質量は不自然に小さい。大統一理論のエネルギースケールを Λ とした場合、ヒッグスの質量二乗 m_h^2 を計算する際に Λ^2 程度の二次発散の量子補正 δm_h^2 が入る：

$$m_h^2 = m_{h0}^2 + \delta m_h^2, \qquad \delta m_h^2 \sim \Lambda^2 \tag{2.8}$$

ただし、m_{h0} はヒッグスの裸の質量 (bare mass) である。Λ が GUT の統一スケールとされている 10^{16} GeV とした場合、28 桁程度の数字が打ち消し合うことになる。この打ち消し合いを標準模型で起こすには、この計算に関係する標準模型のパラメータが微調整 (fine-tune) されている必要があり、これは不自然である。この微調整の不自然さはそもそもヒッグスの質量および電弱対称性の自発的破れがなぜ GUT の統一スケール程度ではなく、100 GeV 程度で起こるかという問題に起因する。この問題を一般的に “階層性問題 (hierarchy problem)” と呼ぶ。階層性問題を解決するためには、二次発散する量子補正が原理的に消えるように標準模型を拡張する必要があるとされている。

2.2 標準模型を超える物理

2.1.2 節で説明したように、素粒子の標準模型は未だに数多くの問題を抱える。これらの問題を解決するべく、様々な標準模型を超える物理が提唱されてきた。ここでは LHC-ATLAS 実験で盛んに探索されている超対称性および余剰次元という 2 種類の BSM 物理についてまとめる。

2.2.1 超対称性

1970 年代に積極的に提唱された超対称性 (Supersymmetry, SUSY) は 50 年経った今も、BSM 物理の有力候補である。端的に説明すると、超対称性とはボソンとフェルミオンの間の対称性である。標準模型のようなゲージ対称性を持つ理論はゲージ変換に対して不変であるが、超対称性を持つ理論は超対称変換に対して不変である[*6]。超対称変換の生成子 Q は次に示すように粒子場のスピンを 1/2 だけ変えることで、ボソンからフェルミオンを、フェルミ

[*5] 一見ニュートリノがそのような候補になりえそうだが、ニュートリノが暗黒物質であると仮定した場合、銀河の形成が説明できなくなってしまうことが知られている [3]。

[*6] ただし超対称性は時空の対称性であるため、厳密にはゲージ対称性とは異なる性質を持つことに留意する必要がある。

オンからボソンを生成することができる：

$$
\begin{aligned}
Q|\text{boson}\rangle &= |\text{fermion}\rangle \\
Q|\text{fermion}\rangle &= |\text{boson}\rangle
\end{aligned}
\tag{2.9}
$$

すなわち超対称性を持つ理論では、フェルミオンとボソンが 1 つの超多重項 (supermultiplet) を形成する。このようにして超対称性を用いて標準模型を拡張するような理論は多く存在するが、その中でも最小超対称標準模型 (Minimal Supersymmetric Standard Model, MSSM) は最も単純であり、頻繁に議論の対象となる。MSSM では、表 2.1 および表 2.2 に示したそれぞれの SM 粒子と対をなす、超対称粒子が導入される。MSSM の代表的な超対称粒子を表 2.3 に挙げる。

表 2.3　超対称粒子の一覧。ここでは電弱対称性の自発的な破れが起こる前の $SU(2)_L$ ゲージボソンを弱ボソン、$U(1)_Y$ ゲージボソンを B ボソンとした。SUSY 粒子は対応する SM 粒子の表記にチルダを付けて表される。

SM 粒子	SM 粒子のスピン	対応する SUSY 粒子	SUSY 粒子のスピン
クォーク (q)	1/2	スクォーク ($\widetilde{q}$)	0
レプトン (l)	1/2	スレプトン ($\widetilde{l}$)	0
グルーオン (g)	1	グルイーノ ($\widetilde{g}$)	1/2
弱ボソン (W)	1	ウィーノ ($\widetilde{W}$)	1/2
B ボソン (B)	1	ビーノ ($\widetilde{B}$)	1/2
ヒッグスボソン (h)	0	ヒグシーノ ($\widetilde{H}$)	1/2

MSSM の主な特徴は、超対称粒子の導入に加えて、新たなヒッグス二重項の導入が必要なことである[*7]。標準模型ですでに導入されていた $Y = +1$ を持つヒッグス二重項 H_u に加え、MSSM では新たに $Y = -1$ を持つヒッグス二重項 H_d を導入する：

$$
H_u = \begin{pmatrix} H_u^+ \\ H_u^0 \end{pmatrix}, \qquad H_d = \begin{pmatrix} H_d^0 \\ H_d^- \end{pmatrix}
\tag{2.10}
$$

すなわち MSSM ではヒッグスに関係する 8 つの自由度が導入される。このうち 3 つはヒッグス機構によって標準模型のゲージボソンによって吸収され、5 つの自由度が残る。つまり MSSM では SM のヒッグス h に加え、新たに 4 つのヒッグスボソン $H^0, H^\pm, A^0$ が導入されることになる。これらは BSM ヒッグスとも呼ばれ、LHC-ATLAS 実験でも探索されている。

他方で、2 つのヒッグス二重項と対をなす超対称ヒッグス二重項 $\widetilde{H}_u, \widetilde{H}_d$ も存在する。電弱相互作用の 4 つのゲージボソンと対をなす超対称粒子 (電弱ゲージーノ) はすべて質量を持つので、超対称ヒッグス二重項の 8 つの自由度のうち 4 つはこれらに吸収される。残る 4 つの自由度はヒッグスと対をなす超対称粒子であるヒグシーノとして残る。これらのヒグシーノは電弱ゲージーノと混合し、4 つの電気的に中性なニュートラリーノ $\widetilde{\chi}_1^0, \widetilde{\chi}_2^0, \widetilde{\chi}_3^0, \widetilde{\chi}_4^0$ および 4 つの電荷を持つチャージーノ $\widetilde{\chi}_1^\pm, \widetilde{\chi}_2^\pm$ という質量固有状態を作る[*8]。

MSSM には R パリティという対称性が課される。R パリティの値 P_R はバリオン数 B、レプトン数 L、およびスピンの大きさ s を用いて、

$$
P_R \equiv (-1)^{3(B-L)+2s}
\tag{2.11}
$$

と定義される。これによると SM 粒子は $P_R = +1$ を、SUSY 粒子は $P_R = -1$ を持つ。R パリティが課されると、粒子の相互作用の前後ですべての粒子の P_R の積が保たれねばならない。すなわち MSSM では次に挙げるようなことが現象論的に起こるとされている：

[*7] この理由は主に 2 つある。1 つ目は超対称性理論におけるラグランジアンの書き方の関係上、標準模型でしていたように 1 つのヒッグス二重項を使って $SU(2)_L$ 二重項の下に属するフェルミオンに質量を与えられなくなってしまうからである。2 つ目の理由は、ヒグシーノのハイパーチャージの寄与によって量子アノマリーが生じてしまうからである。これらの問題は $Y = -1$ を持つ新たなヒッグス二重項を導入することで解決される。

[*8] これらの番号は質量が小さい順につけられる。例えば $\widetilde{\chi}_1^0$ が最も軽いニュートラリーノである。

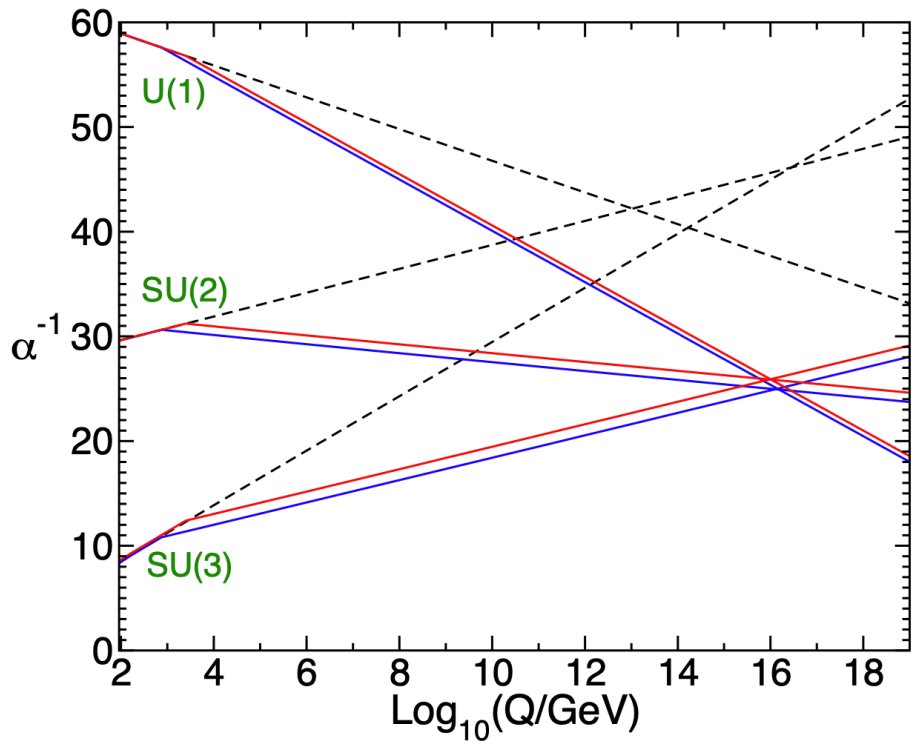


図 2.1　MSSM におけるゲージ結合定数の統一 [4]。2 ループのくりこみ群方程式を用いて計算した、エネルギースケール Q における 3 つのゲージ結合定数の逆数 $\alpha_a^{-1}(Q)$ の大きさを表している。点線は標準模型の場合を表し、実線は MSSM の場合を表す。青い実線は SUSY 粒子の典型的な質量が 750 GeV の場合で、赤い実線は SUSY 粒子の典型的な質量が 2.5 TeV の場合である。標準模型ではいまいち 3 つのゲージ結合定数が一致しないが、MSSM では 10^{16} GeV あたりで良く一致している。

- SUSY 粒子が崩壊する場合は、必ず奇数個の SUSY 粒子が生成される。
- 最も軽い SUSY 粒子 (Lightest Supersymmetric Particle, LSP) は崩壊しない。
- 加速器実験で SM 粒子から SUSY 粒子が生成される場合は、必ず偶数個の SUSY 粒子が生成される。

特に LSP は崩壊先がないため安定である。すなわち LSP が電気的に中性であれば SM 粒子とはほとんど相互作用しないので、これは暗黒物質の有力な候補になる。このようにして超対称性理論は暗黒物質の候補を与えうるので、LHC-ATLAS 実験でも積極的に探索が進められている。

超対称性を導入することの恩恵は暗黒物質の候補を与える以外にもある。まず、大統一理論をより動機付けることになる。図 2.1 に示すように、MSSM では 3 つのゲージ結合定数の一致の度合いが標準模型の場合よりも遥かに良い。また超対称性を導入することで、ヒッグスの質量を計算する際に出てくる 2 次発散が自動的に打ち消し合うようになり、階層性問題も改善される方向に向かう。以上のことから超対称性は有力な BSM 理論として、現在も LHC-ATLAS 実験で広く探索されている。

2.2.2　余剰次元

我々が知る世界は 1 つの時間次元と 3 つの空間次元からなる 4 次元時空であるとされてきた。しかしながらさらなる空間次元の存在を考えることで、標準模型が抱える問題を解決できる可能性がある。これらのような 4 次元よりも高次の時空を扱う理論を一般的に余剰次元理論という。

余剰次元理論の主たる動機は階層性問題の解決にある。弱い相互作用の典型的なエネルギースケールが $O(100\ \mathrm{GeV})$ であるのに対して、4 次元時空における重力の典型的なエネルギースケールはプランク質量 $M_{\mathrm{Pl}} \sim 10^{18}$ GeV である。しかし余剰次元を導入することで、本質的な重力のフラックスが余剰次元方向に逃げ、4 次元時空から見た重力は見かけ上小さいだけであると考えることができる。余剰次元方向も考慮した本質的な重力は実際には強く、弱い相互作用との乖離は元からないとすることで、階層性問題が解決される。以下に 2 種類

の余剰次元模型について簡単にまとめる。

Arkani-Hamed–Dimopoulos–Dvali (ADD) 模型 [5] では、標準模型に含まれる 3 つの相互作用は 4 次元時空のブレーンに閉じ込められており、重力だけ別のコンパクトな空間次元との間を伝播すると考える。例えば半径が R 程度にコンパクト化された余剰次元が n 個あると考えた場合、$(4+n)$ 次元時空におけるプランク質量 $M_{\rm Pl}^{(4+n)}$ と 4 次元時空におけるプランク質量 $M_{\rm Pl}$ の間には、

$$M_{\rm Pl}^2 \sim \left(M_{\rm Pl}^{(4+n)}\right)^{2+n} R^n \tag{2.12}$$

という関係が成り立つ。すなわち余剰次元の大きさ R を大きくすることで、より基本的な重力スケールである $M_{\rm Pl}^{(4+n)}$ を小さく保つことができる。$M_{\rm Pl}^{(4+n)} \sim 100$ GeV、すなわち基本的な重力スケールが電弱スケール程度であると仮定すると、$R \sim 10^{32/n-18}$ m が得られる。$n=1$ とすると R が極端に大きくなってしまい、ニュートン重力が長距離でも成り立たなくなってしまうのでこれは除外される。すなわち ADD 模型では余剰次元の数は $n \geq 2$ と考えられる。

Randall–Sundrum (RS) 模型 [6] では、5 次元の曲がった時空[*9]を考えることで階層性問題を解決する。この模型では重力が強いような 4 次元の “プランクブレーン” から標準模型が住む “TeV ブレーン” に重力が伝播する際に、曲がった時空の寄与による指数関数的な因子によって重力が減衰すると考える。重力は本質的には強いが、我々が住む 4 次元時空においては弱く見えるというロジックは ADD 模型と同じである。しかし余剰次元は 1 次元だけであり、それが極端に小さくても良いという点で ADD 模型とは大きく異なる。

ADD 模型と RS 模型はどちらも余剰次元から標準模型が住む 4 次元時空に重力を伝播させる、Kaluza–Klein (KK) グラビトンというスピン 2 の粒子を予言する。すなわち LHC では KK グラビトンを仮定した余剰次元探索が進められている [7]。これに加え、ミクロなブラックホールの生成を仮定した余剰次元の探索も行われている [8]。

[*9] より正確には 5 次元の anti-de Sitter 空間 (AdS_5) という、負の曲率を持つローレンツ多様体を考える。

第 3 章

LHC-ATLAS 実験

LHC-ATLAS 実験はスイスのジュネーブ市郊外を拠点とする CERN で行われている、大型の国際共同実験である。本章では LHC-ATLAS 実験で使われる LHC 加速器と ATLAS 検出器の概要をまとめる。

3.1 LHC 加速器

Large Hadron Collider (LHC) は、スイスとフランスの国境をまたいで建設された、陽子陽子衝突型の世界最大の円形加速器である。そのビームラインは地下約 100 m に位置しており、周長は 26.659 km ある。図 3.1 に示すように、LHC のトンネルが地上と繋がっている箇所は 8 つあり、それぞれ Point 1 から Point 8 と呼ばれる。そのうちの 4 箇所では陽子衝突が起こるような設計になっており、Point 1 では ATLAS 実験、Point 2 では ALICE 実験、Point 5 では CMS 実験、そして Point 8 では LHCb 実験が行われている。

LHC の第一期運転 (Run 1) は 2010 年から本格的に始まり、重心系エネルギー 7–8 TeV での運転が 2013 年の年始あたりまで行われた。そこから 2015 年までは Long Shutdown 1 (LS1) という初めての長期の運転休止期間が設けられ、加速器や検出器のメンテナンスやアップグレードがされた。2015 年から 2018 年まで行われた第二期運転 (Run 2) では、重心系エネルギーが 13 TeV に上がり、最高瞬間ルミノシティは 2.1×10^{34} cm^{-2}s^{-1} を記録した。そして今現在は 2022 年から開始する予定である第三期運転 (Run 3) に向けた 2 回目の長期運転休止期間 (Long

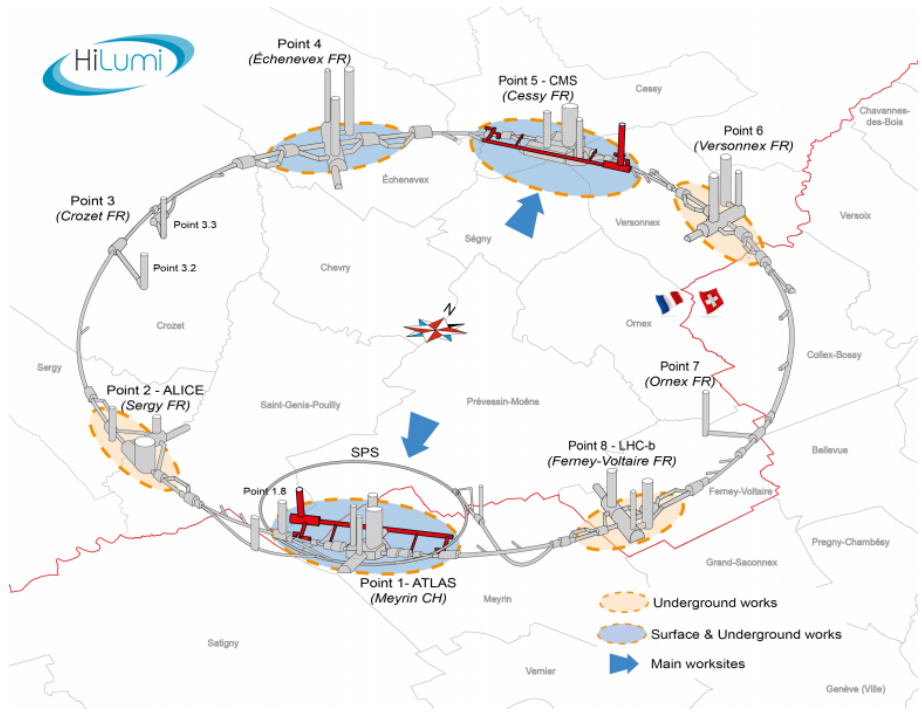


図 3.1　LHC 加速器の俯瞰図 [9]。陽子陽子衝突が起こる地点は Point 1、Point 2、Point 5、および Point 8 である。

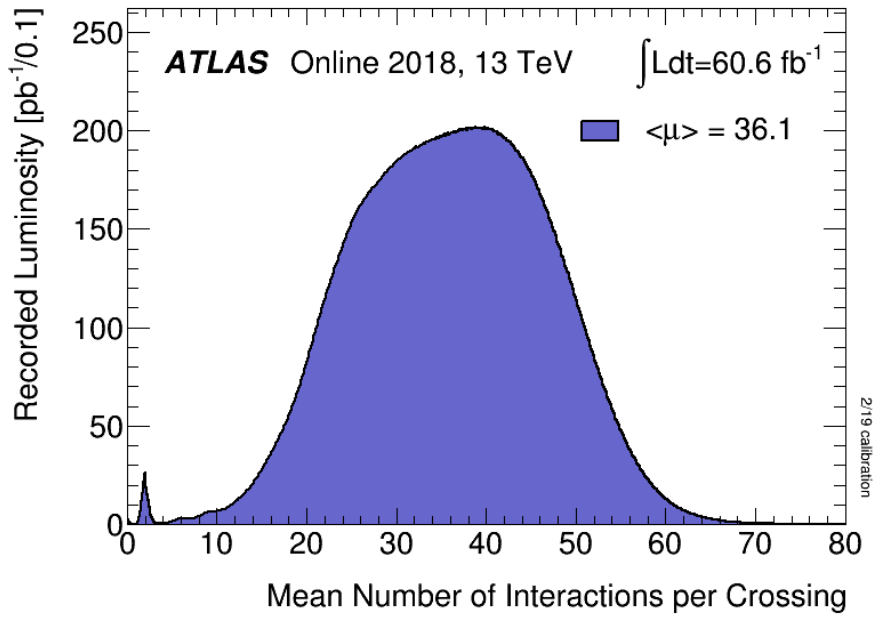


図 3.2　2018 年の各バンチ交差における陽子の平均衝突回数の分布 [10]。$\langle\mu\rangle = 36.1$ は本分布をポアソン分布として見たときの平均値である。

Shutdown 2, LS2) にある。

LHC の陽子ビームは “バンチ” という陽子の集団によって形成され、Point 5 にある radiofrequency (RF) cavity によって 400.8 MHz の高周波でその加速および位相安定化が達成される。この RF cavity に同期するようにバンチが入射される “bucket” が準備される。Bucket は光速 c に近い速度で LHC を周回するので、LHC 1 周に含まれる bucket の数は

$$\frac{400.8\ \mathrm{MHz}}{\frac{c}{26.659\ \mathrm{km}}} \approx 35640 \tag{3.1}$$

となっている。しかし実際はビームの入射とダンプの際の余裕なども考慮して、35640 個の bucket のうち適切に選ばれた 2808 個にバンチが、25 ns の最小時間間隔で入射されるように設計された。すなわち LHC におけるバンチ交差の頻度は $(25\ \mathrm{ns})^{-1} \approx 40.08$ MHz である。入射時にはそれぞれのバンチにおよそ 1.2×10^{11} 個の陽子が入っており、各バンチ交差における陽子陽子衝突の回数は 50–60 回にもおよぶ。図 3.2 に 2018 年の各バンチ交差における陽子の平均衝突回数の分布を示す。また表 3.1 に Run 2 と Run 3 のビームパラメータの比較を示す。Run 3 によって統計量を 2 倍程度に増やす予定である。

表 3.1　Run 2 と Run 3 のビームパラメータの比較。Run 3 が終わる頃には、積分ルミノシティは Run 2 のときのほぼ 2 倍になる。

ビームパラメータ	Run 2	Run 3
重心系エネルギー [TeV]	13	13–14
バンチ交差あたりの陽子の平均衝突回数の最大値	50–60	50–60
最高瞬間ルミノシティ [$\mathrm{cm^{-2}s^{-1}}$]	2×10^{34}	2×10^{34}
積分ルミノシティ*1 [$\mathrm{fb^{-1}}$]	190	350

超伝導電磁石は円形加速器の要である。LHC の陽子ビームの軌道は 1232 個の Nb-Ti の超伝導線材を用いたコイルによる二重極電磁石 (dipole magnet) を使って曲げられている。これらの dipole magnet は加圧超流動ヘリウムを使って 1.9 K まで冷却され、その磁場の大きさは約 8.3 T にもおよぶ。これらの dipole magnet が作る磁場

*1 過去の Run を含めた値。

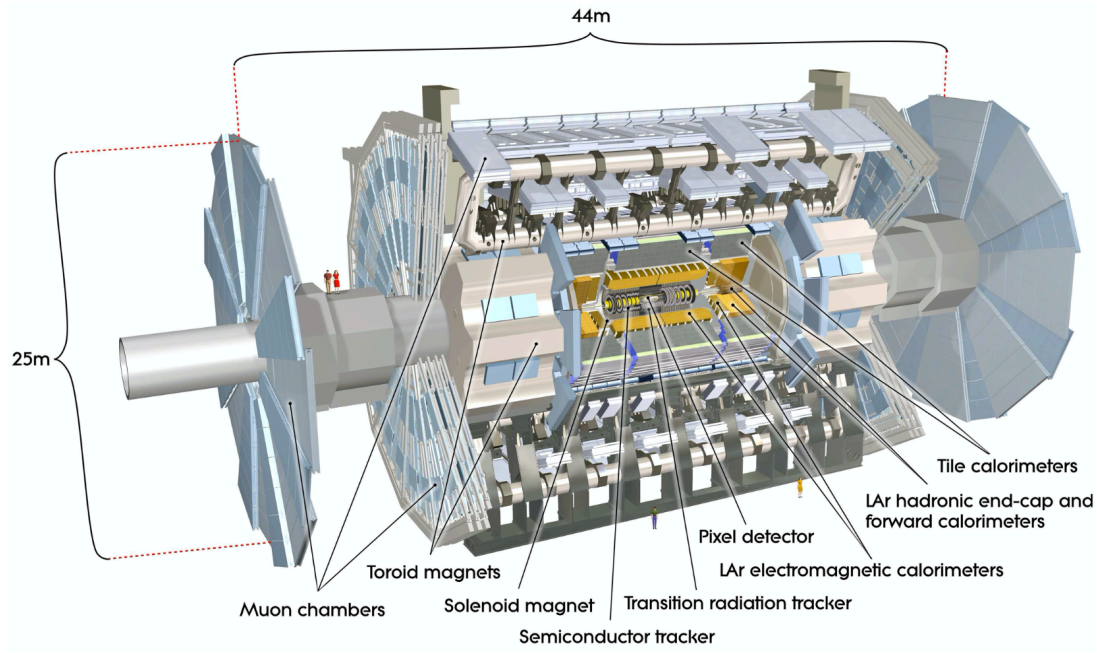


図 3.3　ATLAS 検出器の全体図 [11]。長さ 44 m、直径 25 m、重さ約 7000 トンの円筒型の検出器である。内側から順に内部飛跡検出器、カロリメータ、ミューオンスペクトロメータの 3 つの部分に大きく分けられる。

の補正のために、六重極 (sextupole)、八重極 (octupole)、十重極 (decapole) の電磁石も用いられている。さらに陽子ビーム中の横運動量を持つ粒子をビームのアパーチャーに留めるために、392 個の四重極電磁石 (quadrupole magnet) が用いられている。

3.2　ATLAS 検出器

Point 1 に位置する ATLAS[*2]検出器は大型の汎用型検出器である。図 3.3 にその全体図を示す。ATLAS 検出器に含まれる検出器群は大きく 3 つに分けることができる。最内層にある内部飛跡検出器は荷電粒子の運動量を測定するために用意されている。その外側にあるカロリメータは、電子や光子あるいはハドロンを検出し、それらのエネルギーを測定することができる。そして最外層をなすミューオンスペクトロメータは、内側のすべての検出器を透過してくるミューオンを捉え、その運動量を測定する。本節では ATLAS 検出器が用いる超伝導電磁石や検出器についてまとめる。特に本研究で扱う Level-1 ミューオンエンドキャップトリガーに用いられる TGC 検出器については詳しく説明する。これに加え、事象選別を行うトリガーシステムや LHC のクロック信号を各検出器システムに配布する TTC システムついても説明する。TGC 以外の検出器のより詳しい説明については付録 B を参照していただきたい。

3.2.1　座標系

以降の議論を簡単にするために、まずは ATLAS 実験で使われる標準座標系を紹介する。図 3.4 にその座標系の概要を示す。直交座標系では x 軸は ATLAS 検出器から LHC の円の中心を、y 軸は地上の方向 (鉛直上向き) を指し、原点は検出器中心 (ビーム衝突の中心) にあるような右手座標系を採用している。すなわち z 軸はビーム軸に沿っており、$z > 0$ はジュネーブ市内の方向を、$z < 0$ はフランス側にあるジュラ山脈の方向を指す。$z > 0$ は A-side、$z < 0$ は C-side と呼ばれている[*3]。円筒座標系では、動径を R、方位角を ϕ と定義する。また天頂角 θ を用いて、

[*2] **A T**oroidal **L**HC **A**pparatu**S** (トロイド型 LHC 用実験装置) の略。
[*3] A-side は Airport がある方向、C-side はフランスの Saint-Genis-Pouilly という街にある Charly's Pub というバーがある方向、というように覚えている人が多い。

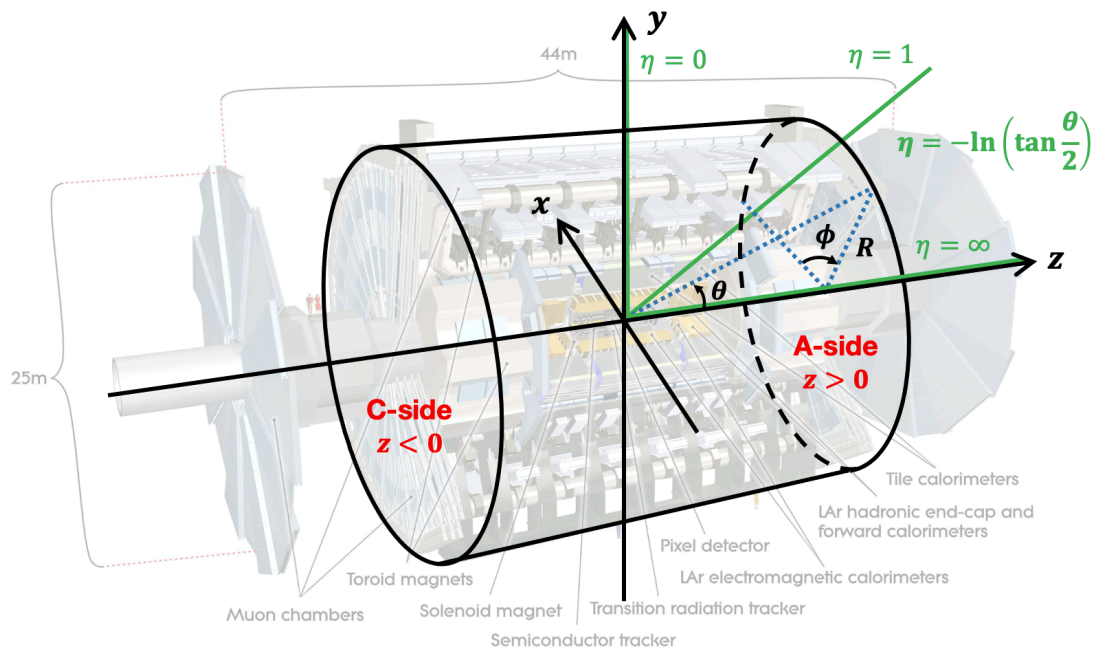


図 3.4　ATLAS 検出器の座標系。x 軸は LHC の円の中心方向を指し、y 軸は地上方向を指すような右手座標系を採用している。$z > 0$ を A-side、$z < 0$ を C-side と呼ぶ。また、η は検出器のカバー領域を表すのに適した座標である。

擬ラピディティ (pseudorapidity) η を

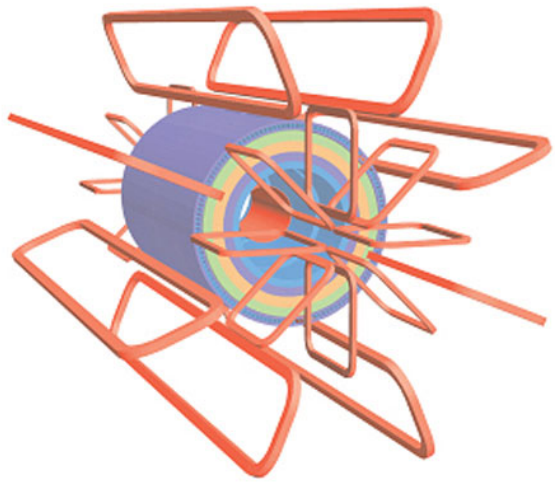
$$\eta = -\ln\left(\tan\frac{\theta}{2}\right) \tag{3.2}$$

と定義する。η は各検出器が覆う領域を説明する際に頻繁に使われる。特に本研究で取り扱うミューオンシステムでは、$|\eta| \lesssim 1.0$ の円筒の側面に対応する領域をバレル (barrel) 部、$|\eta| \gtrsim 1.0$ の円筒の底面に対応する部分をエンドキャップ (endcap) 部と呼ぶ。

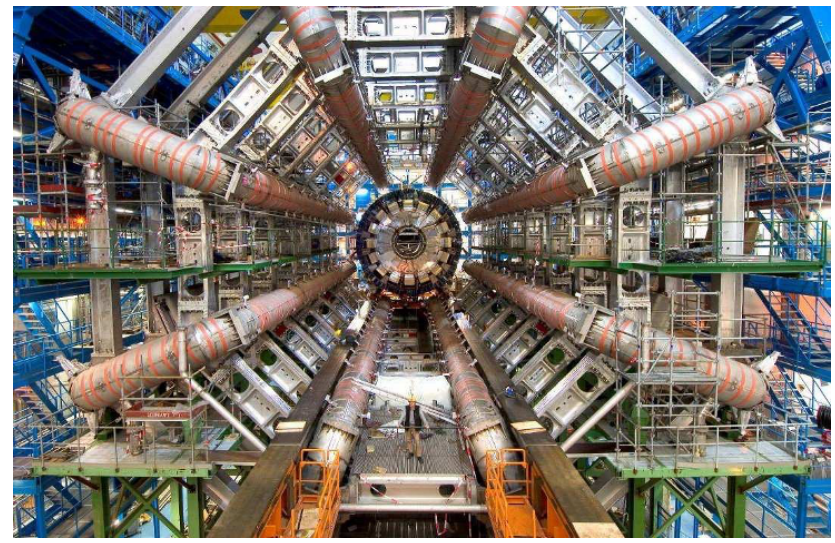
3.2.2　超伝導電磁石

ATLAS 検出器は内部に 3 種類の超伝導電磁石を保有しており、それらを使って荷電粒子の飛跡を曲げ、粒子の識別および運動量の測定を可能にしている。図 3.5(a) にそれらの配置図を示す。最も内部にある central solenoid magnet は、内部飛跡検出器と LAr カロリメータの間にあり、内部飛跡検出器内の粒子飛跡を曲げるために用意されている。その内部の磁場の大きさはおよそ 2 T である。内径は 2.46 m、外径は 2.56 m であり、厚さはわずか 10 cm しかない。これはカロリメータより内側における物質量をなるべく少なくすることで、生成された粒子が無駄にエネルギーを落とすことを避けるような設計になっている。Barrel および endcap toroid magnet は、ミューオンスペクトロメータがミューオンの運動量を測定できるようにするための磁場を作る。それぞれが単体で作る磁場の大きさは約 0.5 T および約 1 T であるが、実際にはこれらが干渉して非一様な磁場が作られる。図 3.5(b) に実際の barrel toroid magnet の写真を示す。

特にミューオンはこれらの 3 種類の超伝導電磁石が作る磁場によって様々な方向に曲げられる。図 3.6(a) および 3.6(b) に ATLAS 検出器内でどのように μ^- の飛跡が曲げられるかの模式図を示す。内部飛跡検出器はすべて central solenoid の磁場領域内にあるので、荷電粒子の飛跡の曲率半径から運動量を測定することができる。ミューオンスペクトロメータはすべてが磁場領域内にあるわけではないので、曲率半径を使った運動量測定ができないものもある。それらでは磁場領域を通過したミューオンの直線飛跡の角度方向を測定することで運動量を概算している。

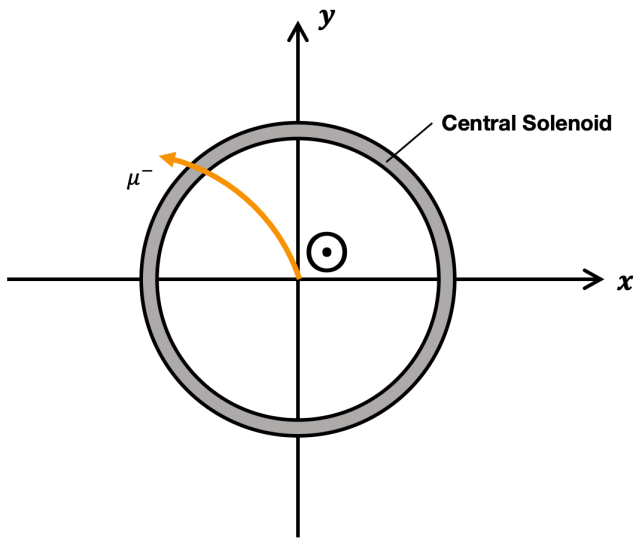


(a) 超伝導電磁石の配置

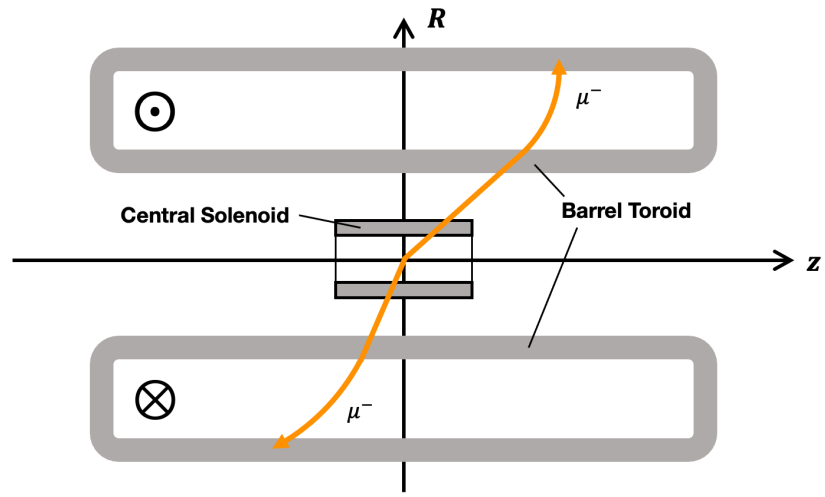


(b) ATLAS 検出器の barrel toroid magnet

図 3.5　ATLAS 検出器の超伝導電磁石 [11]。3.5(a) は 3 種類の電磁石の配置図。Barrel toroid と endcap toroid は互いに重ならないように 22.5° ずらして配置されている。3.5(b) は実際の barrel toroid magnet の写真。真ん中に立っている人の大きさから ATLAS 検出器そのものの壮大さがわかる。



(a) Central solenoid magnet が作る磁場による μ^- の飛跡



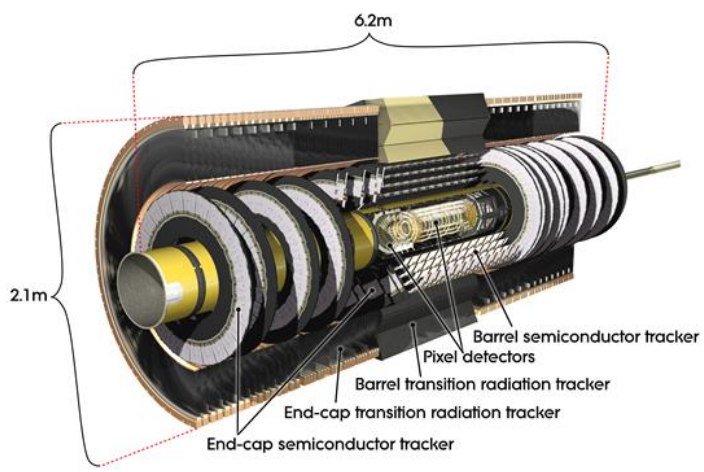
(b) Barrel toroid magnet が作る磁場による μ^- の飛跡

図 3.6　ATLAS 検出器内における μ^- の飛跡。軸の方向はすべて図 3.4 で示したものと同じである。図 3.6(a) では endcap toroid magnet を省略している。

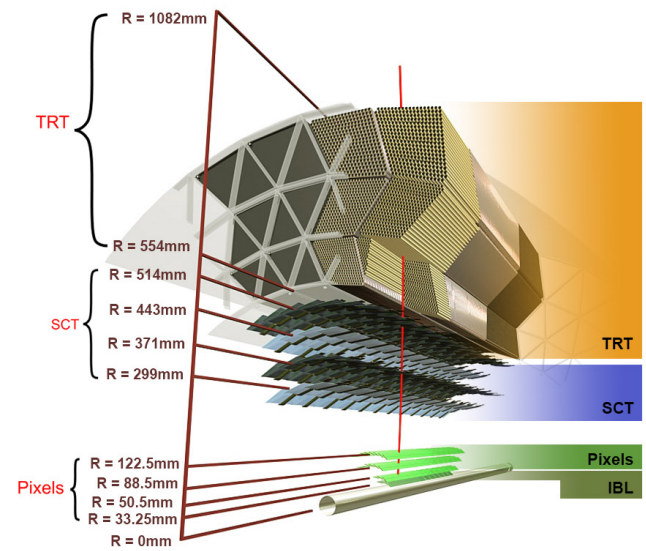
3.2.3　内部飛跡検出器

ATLAS の検出器群のうち最も内側に位置する内部飛跡検出器は、$|\eta| < 2.5$ の範囲における荷電粒子の衝突係数、運動量、および電荷の測定を担っている。内部飛跡検出器は Pixel、Semiconductor Tracker (SCT)、および Transition Radiation Tracker (TRT) の 3 種類の検出器から構成されている。図 3.7 に内部飛跡検出器全体の概要を示す。

Pixel は Run 2 から新たに導入された Insertable B-Layer (IBL) を含む 4 層のシリコンピクセルセンサーからなる検出器であり、高い位置分解能を持つ。SCT は 2 重のシリコンストリップセンサーを 4 層持ち、ステレオ 2 次元測定を用いて荷電粒子のヒット位置を特定する。TRT は多数のドリフトチューブを積み重ねた検出器で、荷電粒子の通過位置測定に加え、遷移輻射の有無によって電子を識別する機能を持つ。これらの 3 つのサブ検出器におけるヒット情報を用いて、central solenoid magnet が作る磁場によって曲げられた荷電粒子の飛跡が再構成される。こ



(a) 内部飛跡検出器の全体図



(b) 内部飛跡検出器層の断面図

図 3.7　内部飛跡検出器の概要 [12, 13]。図 3.7(a) は全体図である。内側から順に Pixel、SCT、TRT 検出器があり、それぞれバレルとエンドキャップの 2 種類に分けられる。図 3.7(b) は検出器層の断面図である。IBL は Pixel 検出器の最内層のことであり、Run 2 から新たに導入された。

れらのサブ検出器の詳細は付録 B.1 に記す。

3.2.4　カロリメータ

ATLAS 検出器のカロリメータの断面図を図 3.8 に示す。ATLAS 検出器のカロリメータはその用途によって、電磁カロリメータとハドロンカロリメータの 2 つに大きく分けられる。電磁カロリメータでは電子と光子の位置およびエネルギーを、ハドロンカロリメータではハドロンジェットなどのエネルギーを測定する。これらのカロリメータは液体アルゴンおよび吸収体として鉛を用いた Liquid Argon (LAr) カロリメータと、シンチレータおよび吸収体として鉄を用いた Tile Calorimeter から構成されている。Level-1 ミューオンエンドキャップトリガーでは、Run 2 から Tile Calorimeter の一部のヒット信号を使用するようになったので、ここでは Tile Calorimeter の層構造についてのみ簡単に説明する。カロリメータの具体的な構造については付録 B.2 で補足する。

図 3.9 に Tile Calorimeter の層の配置を示す。ATLAS 検出器の中心から最も遠い D5 と D6 セルを通過するほとんどの粒子はミューオンであることがわかっている。また、これらのセルは minimum ionizing particle (MIP) の信号をペデスタルから分離するために十分な分解能を有する。よって、これらのセルにおけるヒット情報は Level-1 ミューオンエンドキャップトリガーにおいて偽のミューオン信号を削減するために使われている。

3.2.5　ミューオンスペクトロメータ

ATLAS の検出器群のうち最も外側に位置するミューオンスペクトロメータは、カロリメータを透過してきたミューオンを捕捉し、その運動量を測定するための検出器である。Run 2 まで使われていたミューオンスペクトロメータには Monitored Drift Tube (MDT)、Cathode Strip Chamber (CSC)、Resistive Plate Chamber (RPC)、および Thin Gap Chamber (TGC) の 4 つの検出器が含まれる。MDT と CSC は位置分解能に優れ、運動量の精密測定を担っている。RPC と TGC は時間分解能に優れ、事象選別を行うためのトリガー検出器としての役割を果たしている。この小節では MDT、CSC、RPC については簡単にまとめ、本研究で扱う Level-1 ミューオンエンドキャップトリガーの中心となる検出器である TGC については詳しく見ていく。

図 3.10 と図 3.11 にミューオンスペクトロメータの各検出器の配置を示す。MDT はバレル領域とエンドキャップ

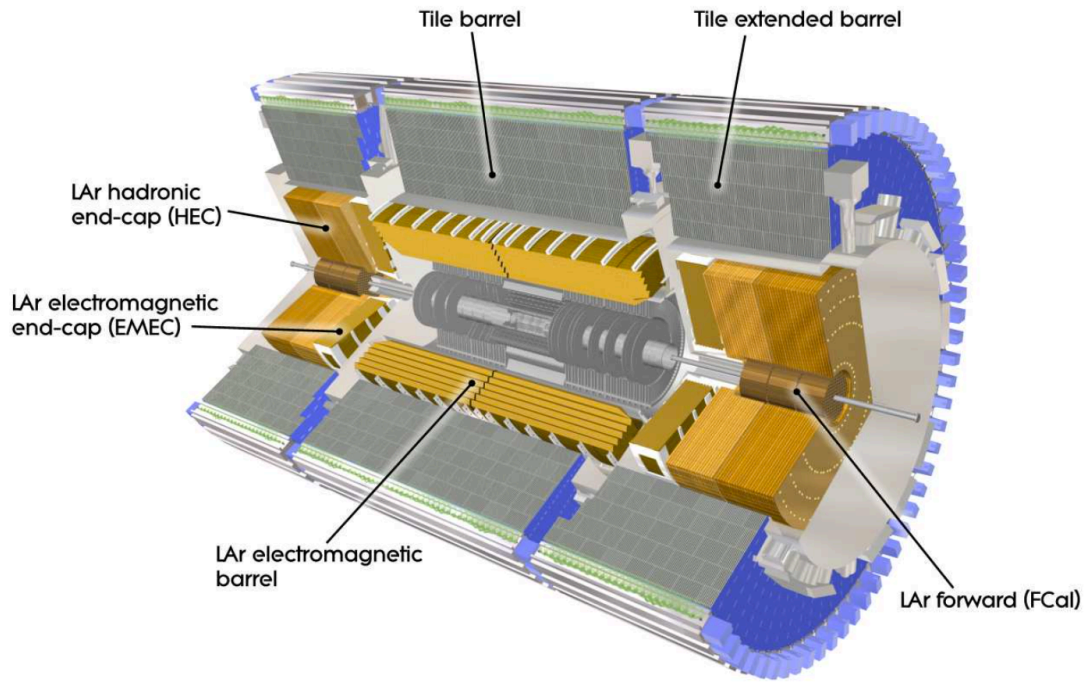


図 3.8　ATLAS 検出器のカロリメータの断面図 [11]。液体アルゴン (LAr) カロリメータは分割されて 3 つのクライオスタットに入っている (バレル 1 つ、エンドキャップ 2 つ)。

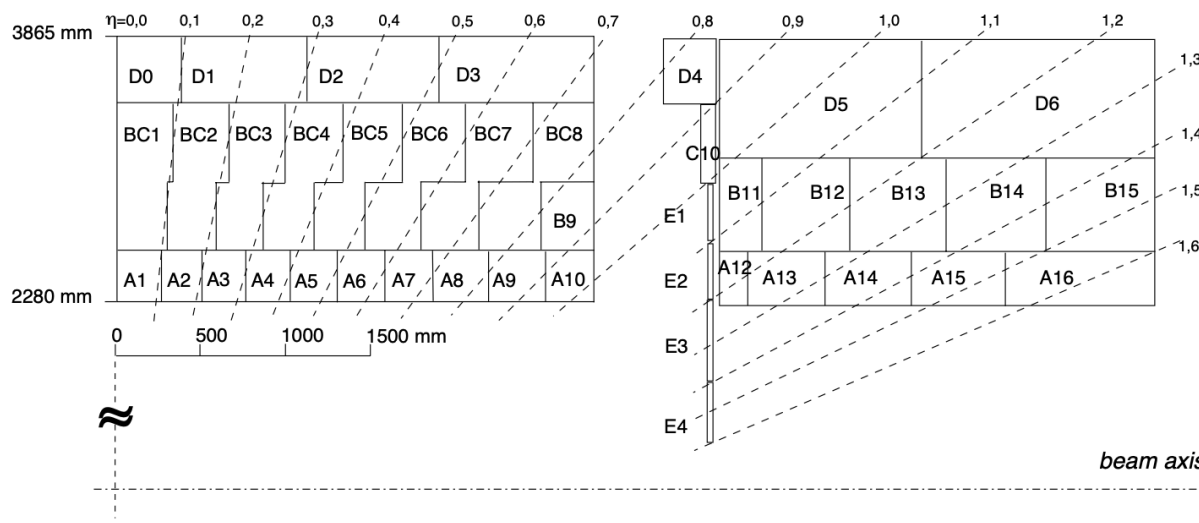
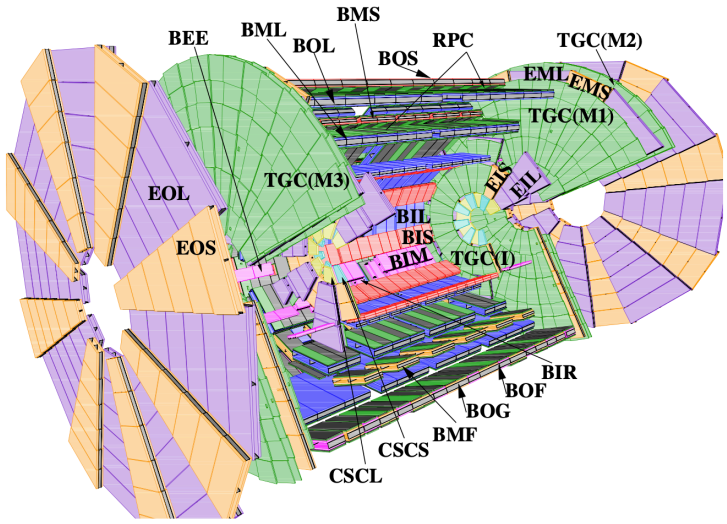


図 3.9　Tile Calorimeter の層の配置 [11]。基本的には R が小さい順に A, B/C, D の 3 層に分かれている。D5 と D6 セルの情報は Level-1 ミューオンエンドキャップトリガーでも使用される。

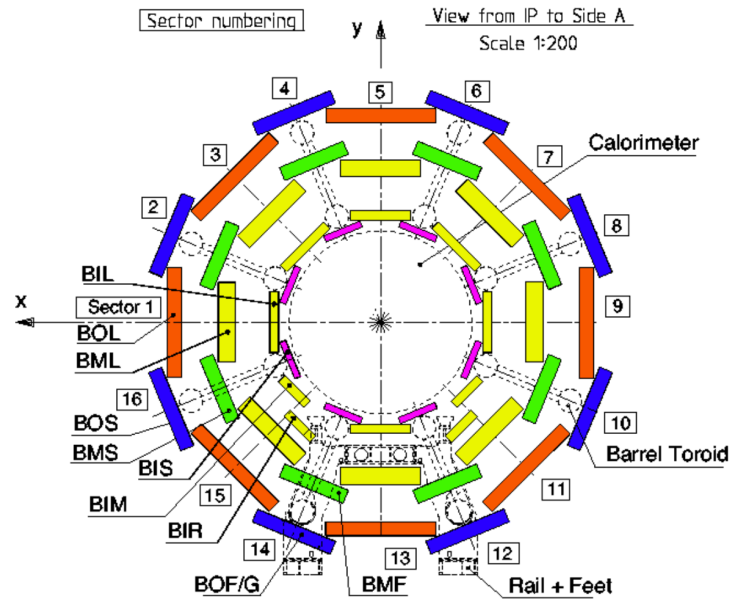
領域の両方に位置する 3 層のドリフトチューブのチェンバーから構成され、ミューオンの運動量の精密測定を行う。CSC はカソードストリップを用いた検出器で、$2.0 < |\eta| < 2.7$ のエンドキャップ領域に位置する。これらの領域では粒子の通過頻度が高いため、それに耐えうる CSC を用いて運動量の精密測定が行われる。RPC は高抵抗のプラスチック板を用いたチェンバーを持つ検出器で、バレル領域の MDT チェンバーに隣接する形で配置される。これらはバレル部におけるミューオントリガー検出器として機能する。MDT、CSC、および RPC のより具体的な説明については付録 B.3 に記す。エンドキャップ領域のミューオントリガー検出器である TGC については以下で詳細を述べる。

Thin Gap Chamber (TGC)

Thin Gap Chamber (TGC) は $1.05 < |\eta| < 2.4$ のエンドキャップ領域をカバーする円盤型のミューオントリガー検出器である。TGC は endcap toroid magnet より内側に位置する Inner (I) と外側に位置する Big Wheel (BW) に大別される (図 3.11 参照)。TGC-I は large sector のみにある MDT の EIL4 チェンバーと隣接している Endcap

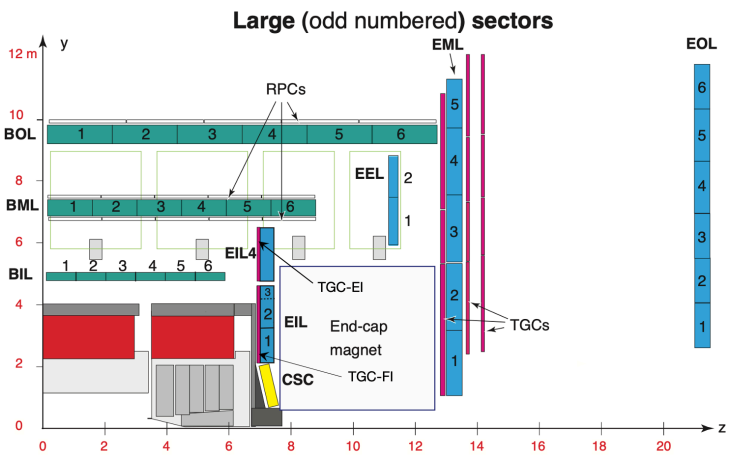


(a) ミューオンスペクトロメータの各検出器の配置と名称

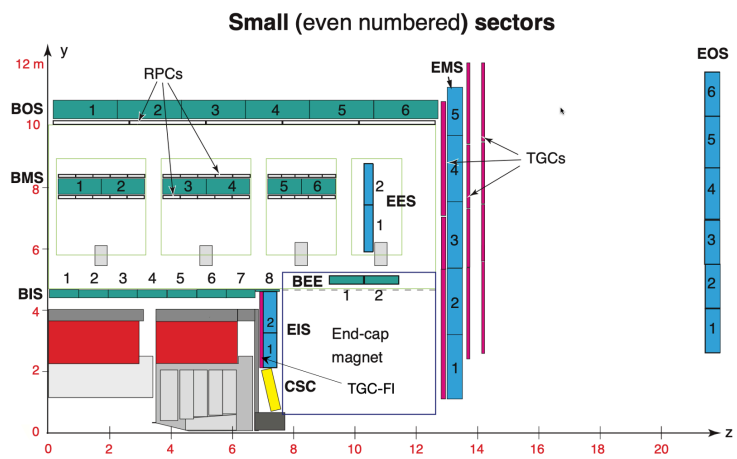


(b) ミューオンスペクトロメータのバレル部の断面図

図 3.10　ミューオンスペクトロメータの各検出器の配置 [11]。図 3.10(a) では各検出器を色分けしており、その名称も記してある。頭文字が B や E から始まっているものはすべて MDT のチェンバーの名称である。図 3.10(b) は z 軸方向から見たバレル部の断面図である。MDT の 3 層の各チェンバーがそれぞれ長方形で示されている。各層には大小 2 種類のチェンバーがあり、大きい方が含まれるセクターを large sector、小さい方が含まれるセクターを small sector と呼ぶ。



(a) Large sector におけるミューオン検出器の配置



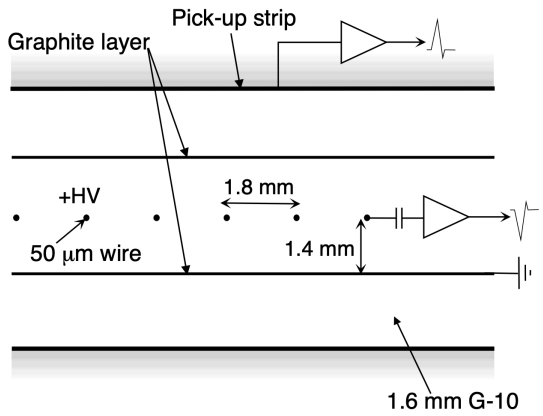
(b) Small sector におけるミューオン検出器の配置

図 3.11　ミューオンスペクトロメータの large sector と small sector における検出器の配置 [14]。図 3.11(a) が large sector の配置を、図 3.11(b) が small sector の配置を表している。

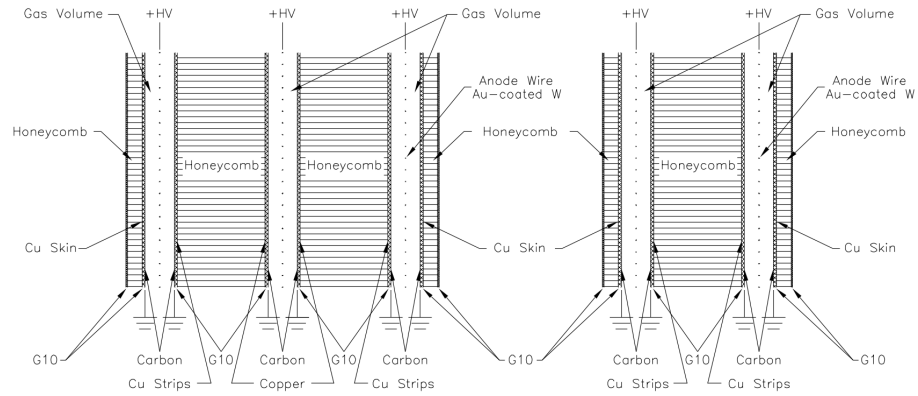
Inner (EI) と、すべてのセクターにある EIS/EIL チェンバーに隣接している Forward Inner (FI) という 2 つのステーションから構成されている。他方で BW は衝突点に近い側から順に M1、M2、M3 という 3 つのステーションから構成されている。M1 と M2 の間には MDT の EMS/EML チェンバーがあるような配置になっている。なお 4.2 節で議論するように、FI ステーションは Run 3 から New Small Wheel という新しい検出器によって置き換えられる予定である。

TGC のチェンバーの構造を図 3.12 に示す。TGC はアノードワイヤー間隔が 1.8 mm、アノードとカソードの間隔が 1.4 mm である MWPC である。2.8 mm のガス層は混合比 55:45 の $CO_2/n\text{-}C_5H_{12}$[*4]で満たされており、その中心には ϕ 方向に直径 50 μm の金メッキされたタングステン製のワイヤーが通っている。ガス層は厚さ 1.6 mm の

[*4] $n\text{-}C_5H_{12}$ とは直鎖状のペンタンのことである。ノルマルペンタンともいう。



(a) チェンバーの断面図



(b) Triplet と doublet ステーションの断面図

図 3.12　TGC のチェンバーとステーションの構造 [11]。図 3.12(a) はチェンバーの断面図。図 3.12(b) は triplet と doublet ステーションの断面図。図の左側が triplet、右側が doublet。

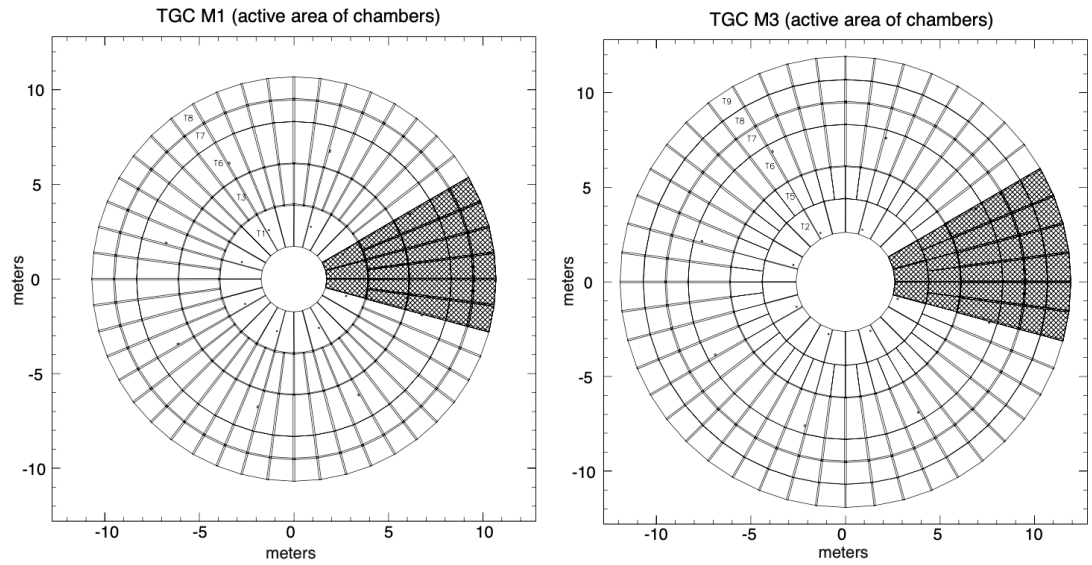


図 3.13　TGC Big Wheel の unit[15]。実線によって区分けされている部分が 1 つの unit である。左側が M1、右側が M3 である。Big Wheel の 3 つのステーションは計 9 種類の unit から作られている。斜線領域は ϕ の 1/8 (octant) 領域である。

G-10(ガラスエポキシ樹脂) に挟まれている。G-10 のガス層側の片面はカーボンが塗布されており、カソード面になっている。もう片方の面は銅で被覆されており、ストリップとして R 方向に分割されている。荷電粒子がガス層を通過すると約 2.8 kV の印加電圧によって電子雪崩が起こり、ワイヤーでは電子雪崩によって生じた正イオンのドリフトが、ストリップではそれらの鏡像電荷が電流信号として検出されるようになっている。

TGC のステーションは 2 層あるいは 3 層のガス層を含んでおり、2 層のものを doublet、3 層のものを triplet と呼ぶ。M1 のみが triplet で、M2、M3、EI、FI はすべて doublet である。図 3.12(b) に示されているように、doublet には 2 層のワイヤーと 2 層のストリップがあり、triplet には 3 層のワイヤーと 2 層のストリップがある。また、ガスチェンバー間はペーパーハニカムによって満たされている。

TGC は台形型の unit と呼ばれる検出器単位から構成されている。図 3.13 に TGC BW の unit の模式図を示す。まず TGC BW は $1.05 < |\eta| < 1.92$ のエンドキャップ領域と $1.92 < |\eta| < 2.4$ のフォワード領域に二分される。エンドキャップ領域は ϕ 方向に 48 分割、フォワード領域は 24 分割されている。さらにエンドキャップ領域は、M1 の場合は $|\eta|$ 方向に 4 分割、M2 と M3 は 5 分割されている。フォワードはどのステーションにおいても各 ϕ について 1 つである。すなわち、M1 は $48 \times 4 + 24 = 216$ 個、M2 と M3 は $48 \times 5 + 24 = 264$ 個の unit から構成されている。

さらに、TGC BW は電気回路制御やデータ読み出し、電源供給の観点から ϕ 方向に 12 個のセクターに分割され

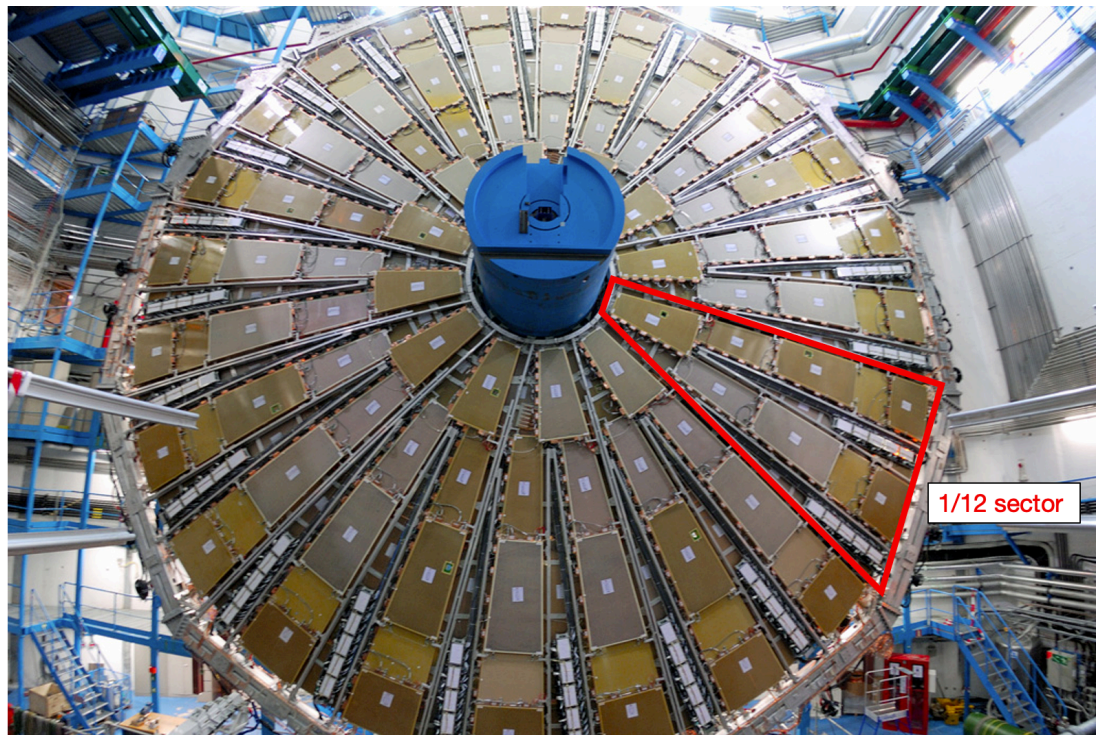


図 3.14　TGC の全体写真と 1/12 セクター [16]。赤枠で囲っている部分が 1/12 セクターである。各 ϕ についてエンドキャップの unit が 4 つあるので、このステーションは M1 であることがわかる。

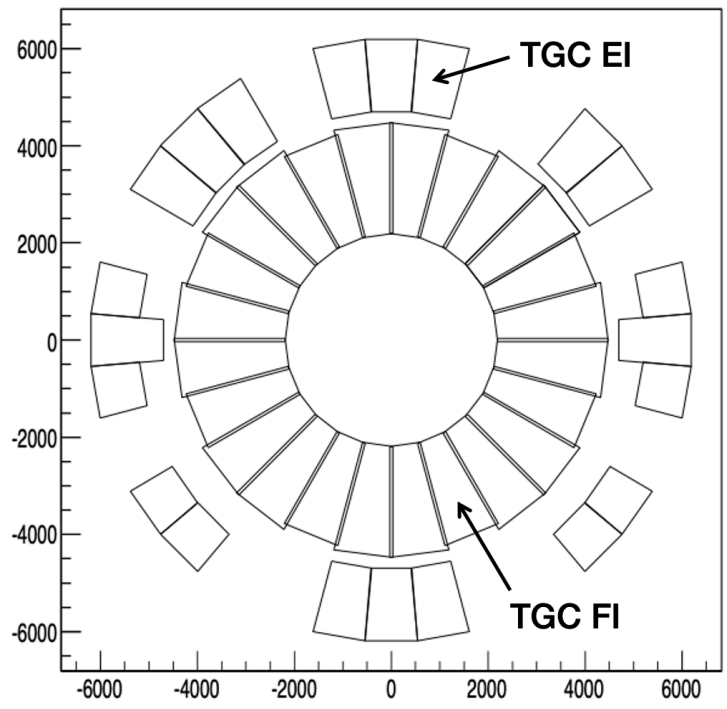


図 3.15　TGC EI と FI のチェンバーの配置 [17]。横軸は x 軸、縦軸は y 軸である。EI は barrel toroid magnet と干渉するためすべての ϕ 領域には設置されていない。

ている。各セクターには x 軸の正の方向から y 軸の正の方向に向かって順に、A/C-01 から A/C-12 までの名前がついている。各 1/12 セクターのエンドキャップ部には M1 の場合は 8 個の unit が、M2 と M3 の場合は 10 個の unit が含まれる。図 3.14 に TGC の 1/12 セクターの写真を示す。

Endcap toroid magnet より内側にある TGC EI と FI のチェンバーの配置を図 3.15 に示す。FI は ϕ 方向に 24 分割されている。EI は barrel toroid magnet と干渉してしまうため、すべての ϕ 領域には設置されていない。EI の unit は ϕ 方向の分割が FI より細かく、計 22 個ある。

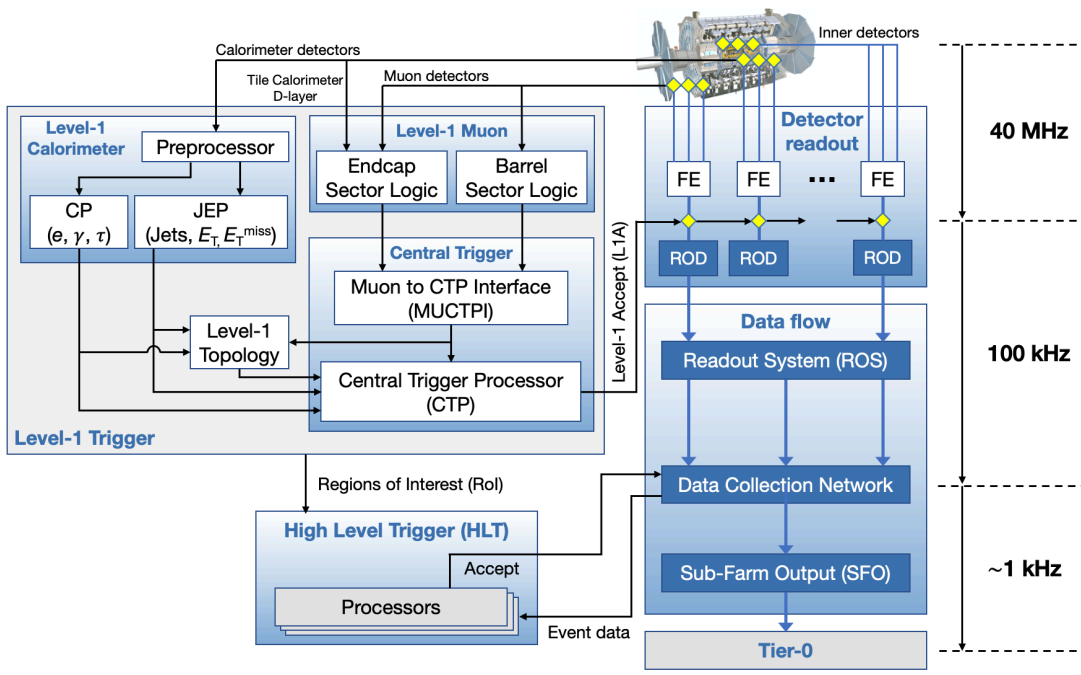


図 3.16　Run 2 における ATLAS の TDAQ システム [18]。図の左側がトリガーシステム、右側が DAQ システムをまとめている。カロリメータやミューオンスペクトロメータの信号を基に、Central Trigger Processor (CTP) が最終的な Level-1 トリガー判定を行う。ある事象データを記録するべきだと判断した場合は Level-1 Accept (L1A) 信号が検出器のフロントエンドなどに送られ、検出器のデータが読み出される。読み出されたデータはさらに HLT によって選別される。

3.2.6　トリガーシステム

3.1 節でも述べたように、LHC では 25 ns 間隔で陽子バンチの交差が起こり、各交差では最大で 50–60 回の陽子衝突が起こる。新物理に関係する現象の断面積は一般的に陽子陽子衝突の全断面積に比べて遥かに小さいので、限られた読み出し帯域とオフラインの計算リソースを最大限有効に活用するためには、興味がある衝突事象のみを記録する必要がある。すなわち検出器の信号のみを頼りに、物理的に重要な衝突事象を短時間で選び出さなければ (トリガーしなければ) ならない。この重要な役割を担っているのがトリガーシステムである。正しい物理データを取得するためには、トリガーシステムとデータ取得 (data acquisition, DAQ) システムが連動して機能する必要がある。ATLAS 実験では、トリガーとデータ取得をまとめて Trigger and Data Acquisition (TDAQ) システムと呼ぶ。図 3.16 に Run 2 における ATLAS の TDAQ システムの概要を示す。ATLAS のトリガーシステムは Level-1 Trigger という初段のハードウェアトリガーと、High Level Trigger (HLT) という後段のソフトウェアトリガーから構成される。

Level-1 Trigger

Level-1 Trigger は初段のハードウェアトリガーであり、イベントレートを 40 MHz から 100 kHz まで削減することが要求される。Level-1 Trigger はカロリメータの信号を基にトリガー判定を行う Level-1 カロリメータ (L1Calo) トリガーと、RPC や TGC の信号を基にする Level-1 ミューオン (L1Muon) トリガーに大別される[*5]。これらの 2 種類のトリガーで処理された情報は最終的に Central Trigger Processor (CTP) に渡され、Level-1 トリガー判定が行われる。ある事象データが後段に渡されるべきだと判断された場合は、L1A 信号が各検出器のフロントエンドなどに送られ、検出器のデータが実際に読み出される。読み出されたデータは各検出器システムの Readout Driver (ROD) でイベントビルディングされ、Readout System (ROS) に送られる。

[*5] 図 3.16 にもあるように Level-1 トポロジカル (L1Topo) トリガーも存在する。これはミューオンやジェットの信号からイベントトポロジーを計算し、その情報を Level-1 のトリガー判定に活用するというものである。Run 2 から新たに導入された。

Level-1 カロリメータトリガーでは、高いエネルギーを持つ電子や光子、ジェット、さらにはハドロンに崩壊する τ の信号を探す。また、検出された粒子のエネルギーから消失横エネルギー (missing transverse energy, MET) $E_{\mathrm{T}}^{\mathrm{miss}}$ を計算し、MET が大きい事象を探すこともできる[*6]。

Level-1 ミューオントリガーは RPC の信号を基にトリガー判定を行う Level-1 ミューオンバレル (L1Muon Barrel) トリガーと、主に TGC の信号を基にトリガー判定を行う Level-1 ミューオンエンドキャップ (L1Muon Endcap) トリガーに分けられる。どちらも Sector Logic というトリガープロセッサーボードによってトリガー判定が行われる。バレルおよびエンドキャップのトリガー判定信号は後段の Muon to CTP Interface (MUCTPI) でまとめられ、CTP に送られる。

陽子陽子衝突によって生成された粒子の信号は L1A が来るまで各検出器のフロントエンドでバッファされる。バンチ交差が起こってからフロントエンドに L1A が届くまでの時間を Level-1 latency (L1 latency) と呼ぶ。L1 latency はトリガーソースとなるサブシステムやイベント中の粒子数などに依存せず、固定の値となるようにデザインされている。すなわち Level-1 Trigger は fixed latency scheme を採用している。この決められた時間内に L1A が来なければ該当するイベントデータは各検出器のフロントエンドで捨てるようになっている。また、バッファの大きさには限りがあるので、設計の段階で L1 latency をなるべく小さくし、システムのデザインを簡素にすることが好ましいとされた。LHC-ATLAS 実験では、Level-1 Trigger は L1 latency が 2.5 μs 以内になるように設計されている。L1 latency のおよそ半分の時間は信号の伝達時間に相当する[*7]。

High Level Trigger (HLT) とデータフロー

High Level Trigger (HLT) は後段にあるソフトウェアトリガーであり、Level-1 Trigger によって選び出された衝突事象から最終的に記録する事象を選別する役割を担っている。Run 2 における HLT が選別した事象レートは平均で 1.2 kHz であり、物理データの記録レートは 1.2 GB/s であった [18]。

図 3.17 に HLT とその後段のデータフローの概念図を示す。HLT は Level-1 Trigger から直接送られてくる Region-of-Interest (RoI) という、興味がある検出器領域の η と ϕ の情報と、Level-1 Trigger によって選ばれたイベントデータを用いてトリガー判定を行う。L1Muon や L1Calo、CTP などから別々に送られてくる RoI 情報はまず RoI Builder (RoIB) というソフトウェアによってまとめられ、後段の HLT Supervisor (HLTSV) に送られる。HLTSV は約 2000 個ある HLT ノードに送られてきた RoI を割り当てて送信する。各 HLT ノードでは 1 つの Data Collection Manager (DCM) および複数個の HLT Processing Unit (HLTPU) というアプリケーションが走っている。HLTSV から送られてきた RoI 情報は DCM によって各 HLTPU に割り当てられる。HLTPU では受け取った RoI 情報を基に、DCM を経由して ROS に必要なイベントデータを部分的に要求する。HLTPU は HLTSV から受け取った RoI と ROS から受け取った部分的なイベントデータのみを用いてトリガー判定を行う。記録すべきと判断されたイベントデータはまとめて Sub-Farm Output (SFO) に送られて一時的に保存され、最終的には永久保存のために CERN のコンピューティングセンターである Tier-0 に送られる。なお HLTSV は各 HLT ノードを監視しており、とあるイベントの処理が完了していたら、ROS にその分のイベントデータを削除するように要求する。このような形で RoI を基点としてイベントデータの重要な部分だけを見ることで、効率的にトリガー判定を下すことができている。

3.2.7 Timing, Trigger and Control (TTC) システム

これまで説明してきた ATLAS 検出器の大部分は LHC と同期して動作しなければならない。ATLAS を含む各実験の検出器を LHC と同期させるために用意されたシステムを CERN 全体で Timing, Trigger and Control (TTC)

*6 暗黒物質候補などの検出器とほとんど相互作用しない粒子は検出器にエネルギーをほとんど落とさないので、もしそのような粒子が生成された事象では、MET が大きくなる。このようにして MET は新物理の発見に向けて重要なパラメータになりうる。

*7 ATLAS 実験室内のフロントエンドから ATLAS 回路室内にあるトリガー用のハードウェアまでのケーブル長は最大で約 100 m あるので、往復にかかる時間は約 1.0 μs である。光信号の伝達時間は 1 m あたりおよそ 5 ns である。

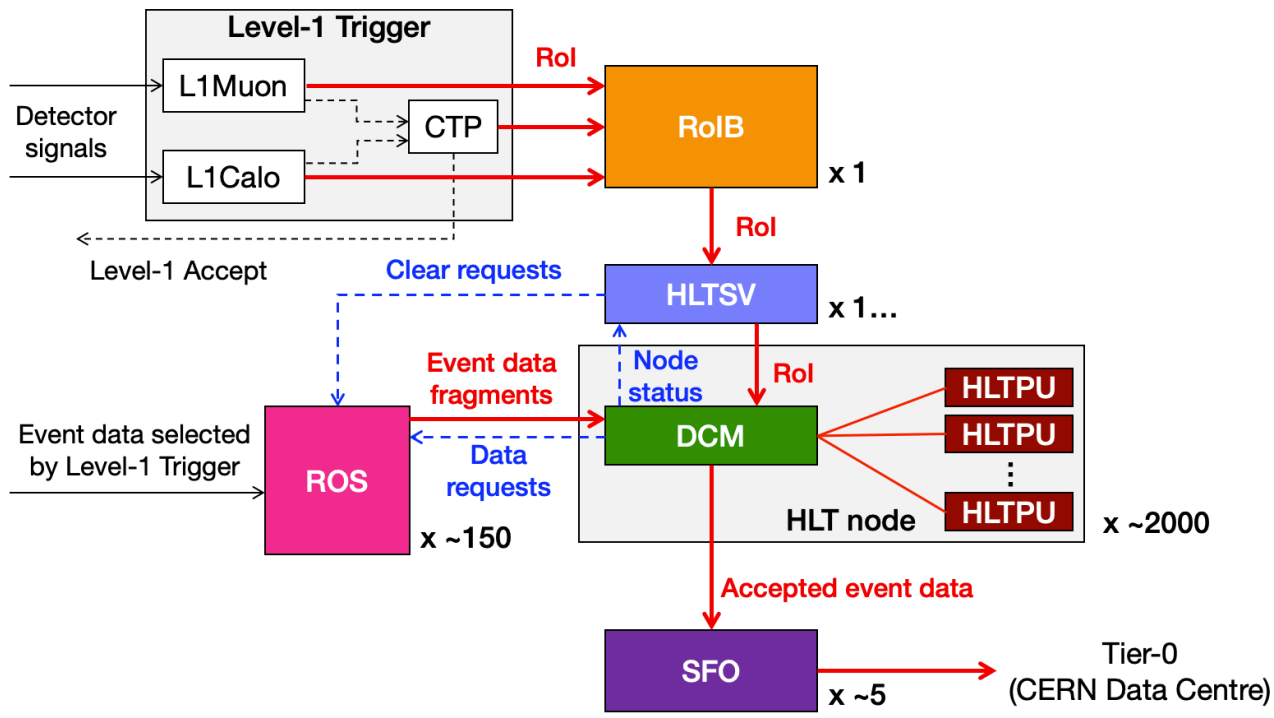


図 3.17　HLT と後段データフローの概念図。HLT は Level-1 Trigger から渡される RoI 情報を基に、イベントデータの重要な部分だけを見てトリガー判定を行うようになっている。

システムと呼び、そのために配布される信号を TTC 信号と呼ぶ。ATLAS では、CTP がすべてのサブシステムに TTC 信号を配布する。TTC 信号には LHC のバンチ交差クロック、Orbit 信号、L1A、Event Counter Reset (ECR) などが含まれる。

バンチ交差 (BC) クロック

3.1 節でも述べたように、LHC では 40.08 MHz の頻度でバンチ交差 (bunch crossing, BC) が起こる。多くの検出器はこの BC の頻度と同じ周期で動作しなければならないため、LHC から 40.08 MHz の BC クロック信号を受信する CTP から各検出器にその BC クロックが配られる。

Orbit 信号

LHC における陽子バンチの最小時間間隔は 25 ns なので、LHC の 1 周あたりに入りうるバンチの最大数は 3564 である[*8]。Orbit 信号はバンチを特定するために LHC から送られてくる 40.08 MHz/3564 = 11.246 kHz の信号である。各サブシステムの TTC システムは Orbit 信号を用いて bunch counter reset (BCR) 信号を生成する。各サブシステムはバンチの数を数え上げ、BCR 信号でその数をリセットすることで 12 ビットのバンチ交差 ID (BCID) を各イベントデータに付与することができる。また Orbit の回数は 24 ビットの Orbit ID を用いて数えられる。

Level-1 Accept (L1A)

3.2.6 節に前述したように Level-1 Accept (L1A) は CTP が出力するトリガー信号である。各検出器のフロントエンドは L1A を受信するとデータを読み出すように設計されている。CTP は L1A とともに、そのイベントの 24 ビットの Level-1 ID (L1ID)、BCID、そしてトリガーの種類を判別するために 8 ビットの trigger type を送る。

Event Counter Reset (ECR)

Event Counter Reset (ECR) 信号は CTP が出力する ATLAS 内部の信号である。取得データを Lumi Block (LB) ごとに区切る際などに使用される。ECR を受けると L1ID は 0 にリセットされる。ECR の回数は 8 ビット

[*8] 3.1 節で議論したように、実際には 35640 個の bucket のうち適切に選ばれた 2808 個に陽子バンチが入射される。

の ECRID によって数えられる。また 8 ビットの ECRID と 24 ビットの L1ID を合わせた 32 ビットは “Extended L1ID” とも呼ばれる。

Busy 信号

Busy 信号は TTC 信号とは別の部類の信号だが、TTC システムの中で重要な役割を持つのでここで説明しておく。Busy 信号とは ATLAS の各検出器システムが新たにデータを受け付けない状態 (busy 状態) にあるときに、CTP へと出力できる信号である。CTP は busy 信号を受信すると一時的に L1A の出力を止め、各検出器のフロントエンドから新たにデータが読み出されないようにすることができる。すなわち各検出器システムは、データ処理が L1A の頻度に間に合っていない場合などにおいて、適切に busy 信号を出力することで読み出しバッファのオーバーフローなどを自ら防ぐ必要がある。

第 4 章

Level-1 ミューオンエンドキャップトリガー

本研究の導入として、本章では Level-1 ミューオンエンドキャップトリガーについて説明する。4.1 節ではまず Run 2 で使われていた現行のシステムについて述べる。次に 4.2 節で、Run 3 に向けて現在進められている Level-1 ミューオントリガーのアップグレード計画について説明する。

4.1 Run 2 における Level-1 ミューオンエンドキャップトリガー

3.2 節でも説明したように、Level-1 ミューオントリガーエンドキャップトリガーは TGC 検出器を中心としたハードウェアトリガーである。本節では Run 2 の Level-1 ミューオンエンドキャップトリガーで使用されたエレクトロニクスやトリガーロジックについてまとめる。まずは導入として TGC Big Wheel (BW) のトリガーセクターやトリガー判定の方法について説明する。

4.1.1 TGC Big Wheel のトリガーセクターと RoI

$1.05 < |\eta| < 2.4$ をカバーする TGC 検出器の BW はトリガーの観点から $1.05 < |\eta| < 1.92$ のエンドキャップセクターと $1.92 < |\eta| < 2.4$ のフォワードセクターに二分されている。図 4.1 に TGC の 1/12 セクターの模式図を示す。各 1/12 セクターには 4 つのエンドキャップトリガーセクターと 2 つのフォワードトリガーセクターが含まれる。1 つのエンドキャップトリガーセクターは η 方向に 37 分割、ϕ 方向に 4 分割され、$37 \times 4 = 148$ 個の小領域に区分されている。この小領域を Region-of-Interest (RoI) と呼ぶ。1 つのフォワードトリガーセクターは η 方向に 16 分割、ϕ 方向に 4 分割され、64 個の RoI に分けられている。

4.1.2 TGC Big Wheel におけるトリガーの概要

TGC BW は M1、M2、M3 の 3 つのステーションでのヒット情報を用いてトリガー判定を行う。図 4.2 にそのトリガーロジックの概念図を示す。陽子陽子衝突によって生成されたミューオンが endcap toroid magnet によって曲げられて TGC の 3 つのステーションを通過することで、各ステーションにおける (R, ϕ) の 2 次元情報が得られる。この 2 次元情報は、M3 におけるヒット点と衝突点を結んだ直線飛跡 (無限大の運動量を持った荷電粒子の飛跡) と M1 や M2 との交点における 2 次元座標と比較される[*1]。これから得られた M1 および M2 における M3 との位置の差分 $(\mathrm{d}R_{13}, \mathrm{d}\phi_{13})$ および $(\mathrm{d}R_{23}, \mathrm{d}\phi_{23})$ を用いて、ミューオンが持つおおよその p_{T} が計算される。この p_{T} に対して閾値を設け、それ以上の p_{T} を持つミューオンがある衝突事象を選別する。各ステーションにおけるコインシデ

[*1] M3 はこのように p_{T} 計算の基準となるので、pivot plane と呼ばれることもある。

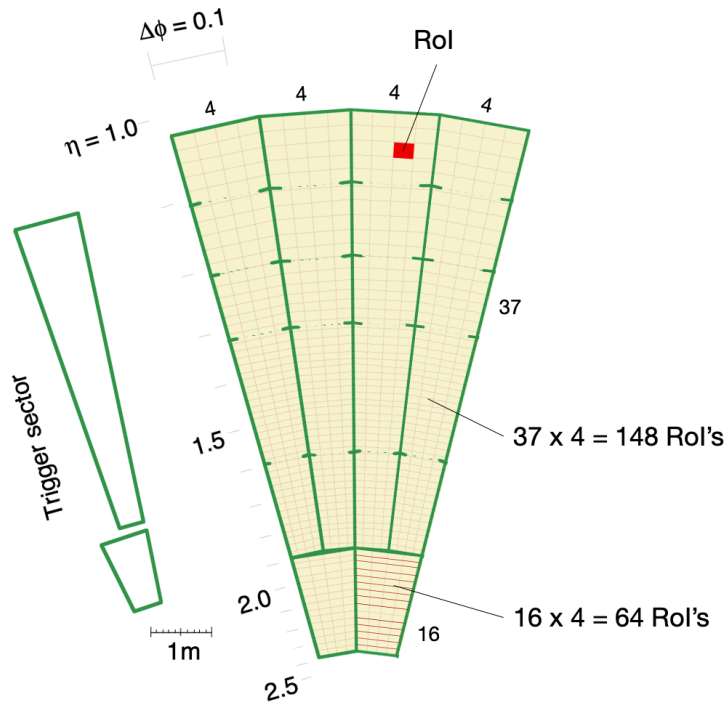


図 4.1　TGC BW の 1/12 セクターにおけるトリガーセクターと RoI [15]。

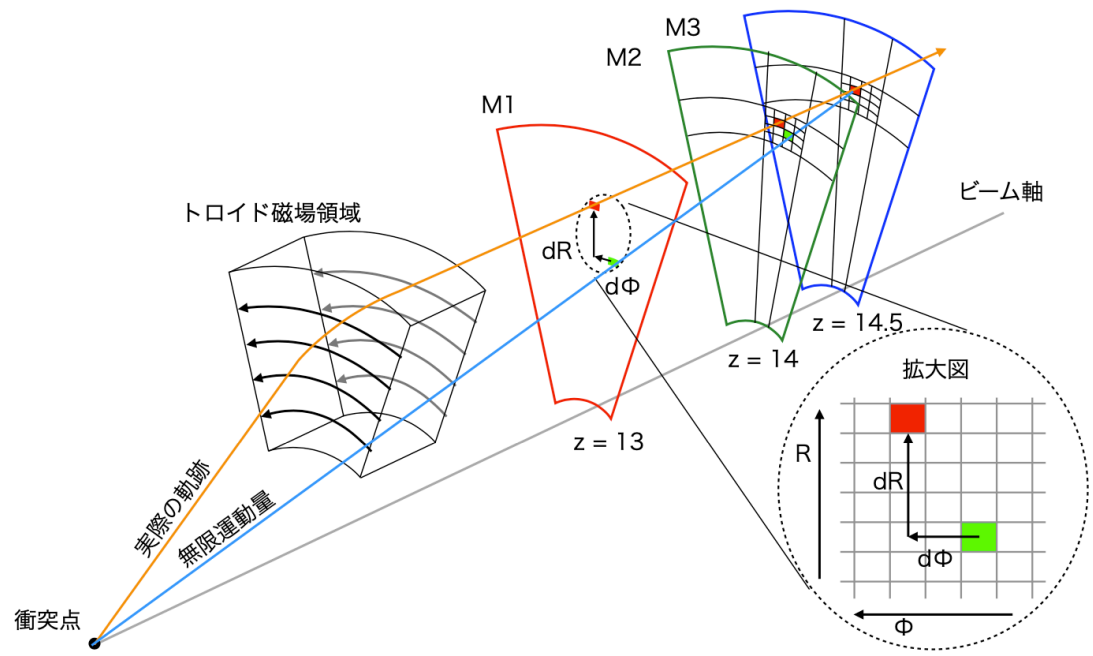


図 4.2　TGC BW のトリガーロジックの概念図 [19]。ミューオンが実際に残すヒット点と直線飛跡の M1 や M2 ステーションとの交点を比較することでミューオンの p_{T} を概算する。

ンスロジックなどの詳細は 4.1.3 節に記述する。

4.1.3　トリガーエレクトロニクスとそのロジック

Run 2 における Level-1 ミューオンエンドキャップトリガーは TGC BW および磁場領域より内側にある TGC EI/FI と Tile Calorimeter からの検出器信号を用いてトリガー判定を行なっていた。図 4.3 にその概要を示す。本小節ではこの図に沿って、トリガーエレクトロニクスとそのロジックについて説明していく。

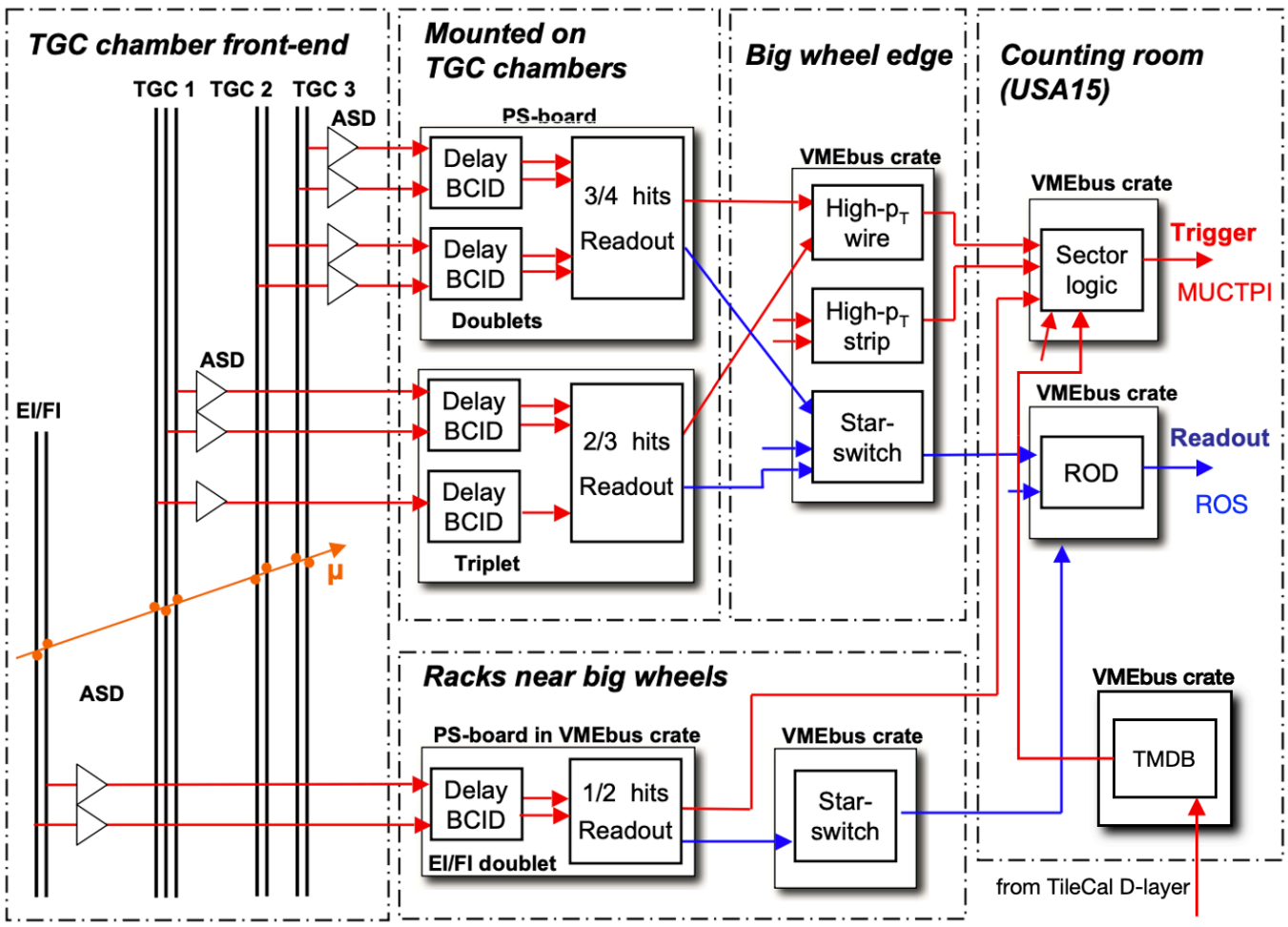


図 4.3　Run 2 における Level-1 ミューオンエンドキャップトリガーのトリガーロジックの概要 [11]。TGC のフロントエンドにあるエレクトロニクスによって TGC BW 内のワイヤー (R) とストリップ (ϕ) のそれぞれでコインシデンスが取られたのち、バックエンドにある Sector Logic ボードによってワイヤー・ストリップ間コインシデンス、および TGC EI/FI と Tile Calorimeter とのコインシデンスが取られる。

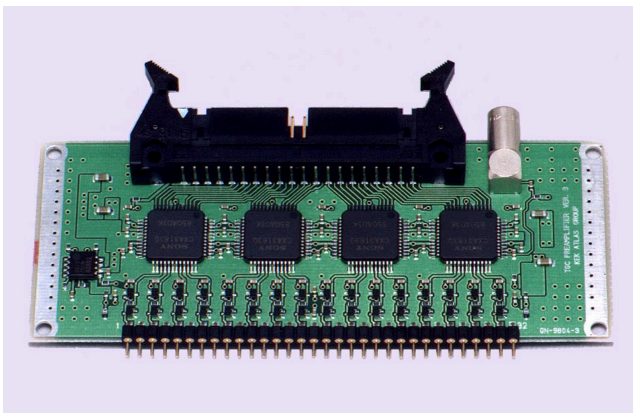
Amplifier Shaper Discriminator (ASD) ボード

TGC のワイヤーとストリップの電流信号はまず Amplifier Shaper Discriminator (ASD) ボードにて電圧信号に変換されたのちに増幅され、最終的には LVDS[*2]規格のデジタル信号に変換されてから後段のフロントエンド回路に渡される。図 4.4 に ASD の概要を示す。1 枚の ASD ボードには 4 枚の ASD チップが載っており、各 ASD チップは 4 チャンネルの増幅と変換を担う。ASD チップの回路は前段増幅器 (preamplifier)、メインの差動増幅回路 (main-amplifier)、コンパレータの 3 段階に大きく分けられる。Preamplifier ではおよそ 0.8 V/pC のゲインで電流信号を電圧信号に変換する。その電圧信号は main-amplifier にて 7 倍に増幅される。最後にその電圧信号はコンパレータにて LVDS 規格のデジタル信号に変換される。また、ASD ボードは試験用に TGC のチャージ出力をエミュレートすることができる[*3]。

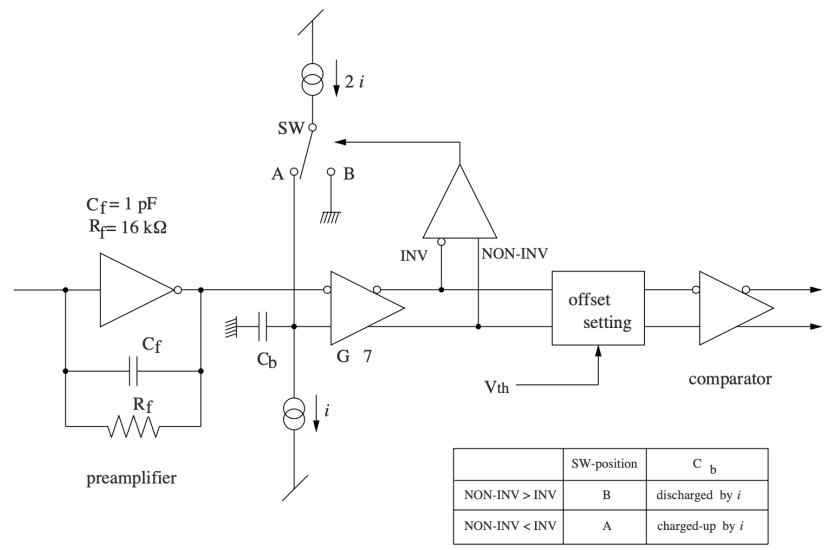
Patch Panel (PP) ASIC

ASD ボードによって LVDS 規格に変換された検出器信号は次に、TGC 検出器上に設置されている PS Board 上にある[*4] Patch Panel (PP) ASIC に入る。1 枚の PP ASIC は 2 枚の ASD ボードから 32 チャンネルの信号を受け取る。各チャンネルからの信号はミューオンの ToF やケーブル長の違いによって、異なるタイミングで PP

[*2] Low Voltage Differential Signaling の略。
[*3] この機能を使って打つテストパルスを ASD Test Pulse という。
[*4] TGC BW の PS Board は M1 と M3 にのみあり、M2 にはない。TGC EI/FI 用の PS Board は検出器上ではなく TGC BW の近くのラックの VME クレートにある。



(a) ASD ボードの写真



(b) ASD チップの回路ブロック図

図 4.4　ASD の概要 [20]。図 4.4(a) は ASD ボードの写真。ボード 1 枚あたり 4 枚の ASD チップが搭載されている。図 4.4(b) は ASD チップの回路ブロック図。ASD チップの回路は電流信号を電圧信号に変換する preamplifier、電圧信号を増幅させる main-amplifier、そして電圧信号を LVDS 規格に変換するコンパレータの 3 段階に大きく分けられる。

ASIC に入ってくる。加えて、ASD の信号と LHC の 40 MHz クロックとの間には位相差がある。すなわち、PP ASIC では TGC の各チャンネルから来た信号のタイミング調節を行い、32 チャンネル単位で同じバンチ交差から来たすべての信号を LHC のクロックと同位相になるように合わせる。この機能をバンチ交差識別 (bunch crossing identification, BCID) と呼ぶ。タイミング調節は ASD ボード単位 (16 チャンネル単位) で行えるようになっている。

Slave Board (SLB) ASIC

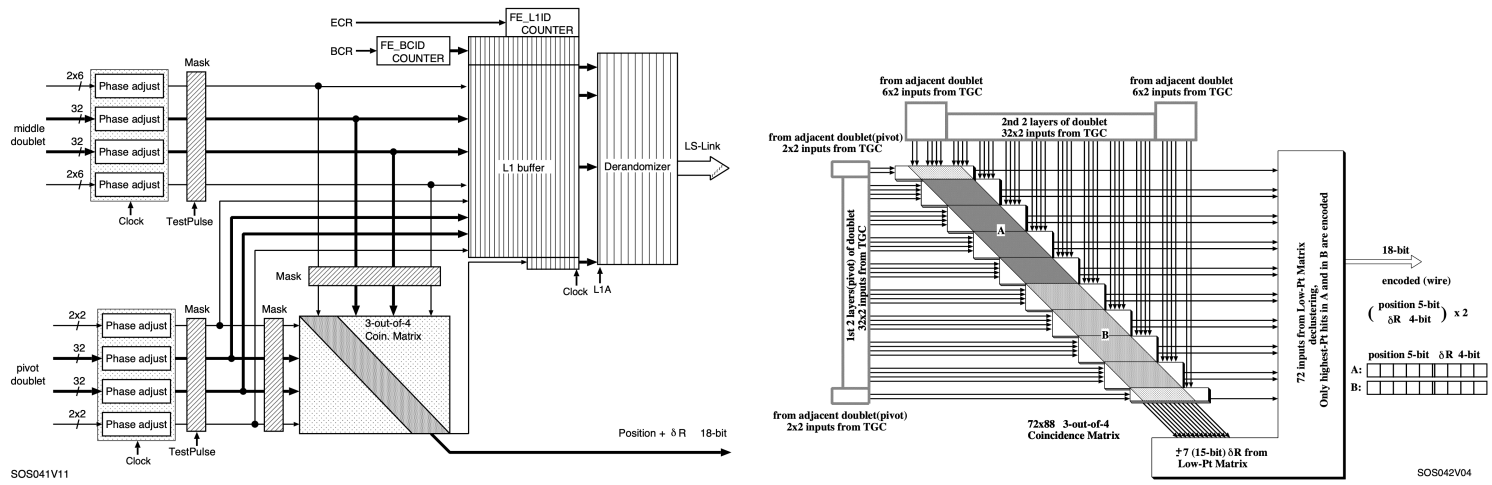
PP ASIC によって BCID された信号は同じく PS Board 上にある Slave Board (SLB) ASIC に入力される。図 4.5(a) に示すように、SLB ASIC は 2 つの役割を持つ：

- ステーション間 (M2 と M3) あるいはステーション内 (M1) コインシデンスを取り、その結果を後段のトリガーロジックボードである High-p_{T} ボードに渡す (トリガーパス)
- 各チャンネルから受信した信号を L1A 信号が来るまで一時的にためておく (読み出しパス)

コインシデンスの取り方の観点から、TGC BW 用の SLB ASIC は 3 種類に分けられる。Doublet 用の SLB ASIC では、M2 と M3 の 4 層分のワイヤーあるいはストリップ信号を受信し、そのうちの 3 層以上にヒットがあることを要求する (3-out-of-4、3/4 コインシデンス)。Triplet ワイヤー用の SLB ASIC では、M1 の 3 層分のワイヤー信号のうち 2 層以上でヒットがあることを要求する (2-out-of-3、2/3 コインシデンス)。Triplet ストリップ用の SLB ASIC では、M1 の 2 層分のストリップ信号のうち 1 層以上でヒットがあることを要求する (1-out-of-2、1/2 コインシデンス)。コインシデンスは図 4.5(b) に示すようなコインシデンス行列を用いて取られる。1 枚の SLB ASIC あたり後段に渡すヒット情報は以下の通りである：

- Doublet 用：最大 2 つの $(R_3, \mathrm{d}R_{23})$ または $(\phi_3, \mathrm{d}\phi_{23})$
- Triplet ワイヤー用：最大 3 つの R_1
- Triplet ストリップ用：最大 4 つの ϕ_1

LVDS シリアル通信を用いて出力するヒット情報は High-p_{T} ボードに送られる。なお TGC EI/FI 用の SLB ASIC では 1/2 コインシデンスが取られたのち、G-Link というシリアル通信規格を用いて Sector Logic ボードに送られ



(a) SLB ASIC の処理のブロック図　　(b) SLB ASIC のコインシデンス行列

図 4.5　Doublet ワイヤー用の SLB ASIC の概要 [15]。図 4.5(a) は SLB ASIC の処理のブロック図。Doublet 用なので 3/4 コインシデンスを要求している。上側が読み出しパス、下側がトリガーパスである。図 4.5(b) は図 4.5(a) の下部にあるコインシデンス行列の概要である。隣り合う領域からのチャンネルも含めた計 160 チャンネルを受信し、4 層間のコインシデンスを取り、後段の High-p_{T} ボードに 2 つのヒット情報を渡す。Doublet ストリップ用の出力は 18 ビットではなく 16 ビットである。

る (図 4.3 参照)。3.2.6 節で述べたようにトリガーパスでは fixed latency でのオペレーションが肝要であるので、それを実現した安定動作を保証する必要がある。そのため、受信側で膨大な量の運転パラメータを正確に設定することや、特別な手続きでのリンクの初期化を行うことが必要不可欠である。

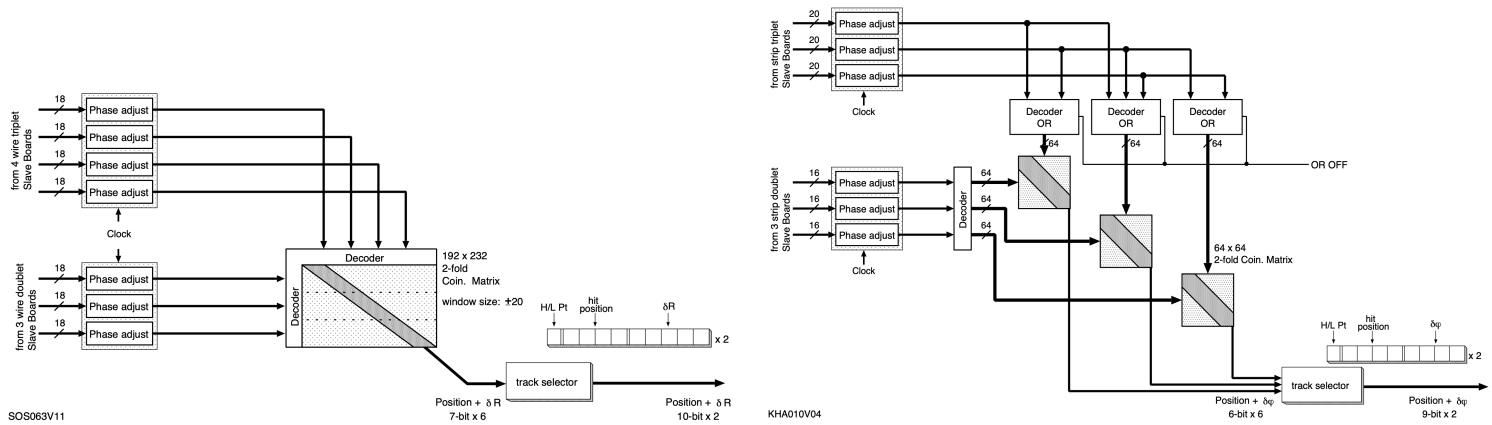
読み出しパスでは、まず Level-1 buffer (L1 buffer) というバッファを用いて L1A が来るまでイベントデータを BCID とともに保持する。L1A が来たらそのイベントおよびその前後 1 BC を合わせた計 3 BC 分のイベントデータを読み出す。読み出す際に、各イベントデータには L1ID が付与される。次に、3 BC 分のイベントデータはトリガーデータとともに derandomizer という、シリアル化待ちバッファに渡される。シリアル化されたデータは LVDS 規格で後段の Star Switch ボードに送られる。

High-p_{T} (HPT) ボード

4.1.4 節に後述する Star Switch と同じクレート内にある High-p_{T} (HPT) ボードは、SLB から来るトリガー信号を基に triplet と doublet 間のコインシデンスを取る役割を持つ。ワイヤーとストリップは別々に、それぞれワイヤー用の HPT ボードとストリップ用の HPT ボードにてコインシデンスが取られる。HPT ボード上には 3 枚あるいは 4 枚の HPT ASIC が搭載されている。図 4.6 に HPT ASIC の概要を示す。HPT ASIC は SLB ASIC と同じようにコインシデンス行列を用いて、triplet SLB と doublet SLB の出力から M1 と M3 の位置の差 $\mathrm{d}R_{13}$ または $\mathrm{d}\phi_{13}$ を計算する。コインシデンス行列では最大で 6 本のトラックが選ばれ、ストリップ用ではそれらの $(R_3, \mathrm{d}R_{13})$ が、ワイヤー用では $(\phi_3, \mathrm{d}\phi_{13})$ の情報が track selector に渡される。Track selector では $\mathrm{d}R_{13}$ または $\mathrm{d}\phi_{13}$ が小さい、つまりは p_{T} が高いトラックが最大で 2 本選ばれる。これらの情報は G-Link 規格を用いた光信号によって後段のトリガー判定ボードである Sector Logic に送信される。HPT と SL の間の通信においても fixed latency での運用を実現するために、正確な運転パラメータの設定とリンク確立の手続きが肝心である。

Sector Logic (SL)

Sector Logic (SL) は USA15 (ATLAS 回路室) にあるバックエンドボードであり、HPT ボードから来る TGC BW のコインシデンス情報と磁場領域より内側の検出器である TGC EI/FI および Tile Calorimeter から受信した情報を用いてトリガー判定を行う役割を担っている。出力したトリガー信号は後段の MUCTPI に渡される。4.2 節に後述するように Run 3 からは SL は New Sector Logic に置き換わるが、ここでは Run 2 まで使用していた SL について説明する。



(a) ワイヤー用の HPT ASIC の概要　　(b) ストリップ用の HPT ASIC の概要

図 4.6　HPT ボードの概要 [15]。図 4.6(a) がワイヤー用、図 4.6(b) がストリップ用である。コインシデンス行列で最大 6 本のトラックが選ばれたのち、track selector によって p_{T} が大きい順に最大 2 本のトラックが選ばれる。

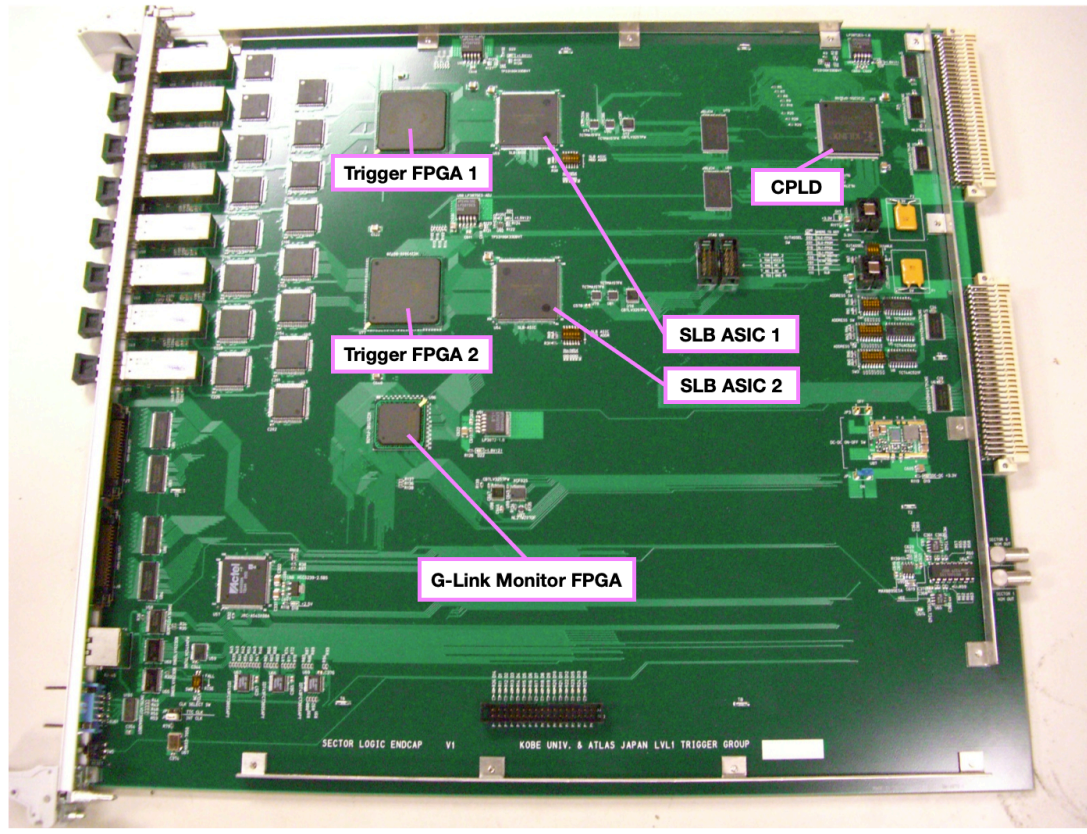


図 4.7　Run 2 で用いたエンドキャップ用 SL の写真 [21]。1 枚のエンドキャップ用 SL にはトリガー判定用の FPGA が 2 枚、読み出し用の SLB ASIC が 2 枚、G-Link 信号監視用の FPGA が 1 枚、そして VME 制御用の CPLD が 1 枚搭載されている。

SL にはエンドキャップセクター用とフォワードセクター用の 2 種類があり、それぞれ 2 つのトリガーセクターを担当する。すなわち、図 4.1 に示す 1/12 セクターは 2 枚のエンドキャップ用 SL[*5] と 1 枚のフォワード用 SL によって担当されている。図 4.7 にエンドキャップ用 SL の写真を示す。トリガー判定は搭載されている各 FPGA[*6] を用いてトリガーセクター別に行うようになっている。

[*5] Level-1 ミューオンバレルトリガーのトリガー判定ボードも Sector Logic と呼ばれるので、Level-1 ミューオンエンドキャップトリガーの SL はそれと区別するために Endcap SL としばしば呼ばれる。ここの議論に使用しているエンドキャップ、フォワードという語句は Endcap SL のエンドキャップセクター用とフォワードセクター用という意味なので、混同しないように注意が必要である。

[*6] Field Programmable Gate Array の略。ユーザーが自由にプログラムできる IC チップであり、電源が入るたびにビットストリームの再読み込みが行われる。

SL はまず、ワイヤー用 HPT から来た $(R_3, \mathrm{d}R_{13})$ とストリップ用 HPT から来た $(\phi_3, \mathrm{d}\phi_{13})$ を用いて R-ϕ コインシデンスを取る。R-ϕ コインシデンスの処理では、(R_3, ϕ_3) から各トラックの RoI を決定し、Look Up Table (LUT) を用いて $(\mathrm{d}R_{13}, \mathrm{d}\phi_{13})$ の情報が p_T に変換される。ここまでの処理は Big Wheel (BW) Coincidence ともいう。BW Coincidence によって得られた RoI と p_T 情報は後段の Inner Coincidence の処理に渡される。

Inner Coincidence は磁場領域より内側にある TGC EI/FI や Tile Calorimeter でのヒットを要求することで、陽子陽子衝突に由来しない荷電粒子によって出力される偽のトリガー信号 (フェイクトリガー) を削減する目的を持つ。TGC EI/FI 用の PS Board 上にある SLB から来る信号と Tile Muon Digitizer Board (TMDB) から来る Tile Calorimeter D 層の信号はともに G-Link 規格の光信号である。Inner Coincidence を経たトラックは 1 つのトリガーセクターあたり、p_T が大きい順に最大 2 本まで選ばれる。これらのトラックの RoI と p_T は Level-1 ミューオントリガーを取りまとめる MUCTPI に渡される。またこれらの選ばれたトラック情報は SL ボード上にある SLB ASIC を経て読み出しパスにも送られる。

Burst Stopper (NIM Process Module)

2012 年の Run 1 の最中、$O(1\ \mu\mathrm{s})$ という短時間の間に L1A が連続して出力されたかつ、それらのイベントにおけるヒットの数が通常より 100 倍も多いような状態が生じた [17]。このような “バースト状態” では TGC を含む複数の検出器システムでトリガー信号が大量に出力されていた。それらのトリガー信号に基づいて CTP が大量の L1A を出力した結果、TGC システムにおいては 4.1.4 節に後述する読み出しシステムのバッファがオーバーフローを起こし、ATLAS 検出器全体のデータ収集を一時的に止めるような状態に陥った。しかしこれらのバースト状態においてはカロリメータでのエネルギー損失が見られず、バーストを起こす検出器セクターに偏りがあったことから、何らかの非物理的なものがその原因であると結論付けられた。すなわち Run 2 が始まるまでにこのようなバースト状態を止める機構を開発する必要があった。

そこで TGC グループは SL のトリガー情報からバースト状態を検知し、トリガーを veto する Burst Stopper (NIM Process Module) というハードウェアを開発した。Burst Stopper は主に次の 2 つの機能を持つ：

- ある一定の時間間隔においてすべての SL が出力するトリガーの回数を監視し、バースト状態を検知した場合は TGC システムが出力するトリガー信号を veto する。
- バースト状態を検知した場合、CTP にその情報を送り、L1A の出力を止めさせる。

バースト状態は TGC 以外の検出器でも起こるので、L1A の連続出力を止めるには後者の機能も必要となる。Burst Stopper は全 72 台の SL のそれぞれから NIM 規格のトリガー信号を受信し、TGC のすべてのトリガーセクターの情報を基にバースト状態を判定する。

Tile Muon Digitizer Board (TMDB)

陽子陽子衝突由来のミューオンでない荷電粒子によるフェイクトリガーを削減するべく、Run 2 からは Tile Calorimeter の extended barrel の D 層の信号が Level-1 ミューオンエンドキャップトリガーに使われるようになった。具体的には $1.05 < |\eta| < 1.3$ の領域で Tile Calorimeter の D5 と D6 セル (図 3.9 参照) の信号が用いられる。これらの信号は USA15 に設置されている Tile Muon Digitizer Board (TMDB) によってまとめられる。Tile Calorimeter は 64 個のモジュールに分割されていて、1 枚の TMDB は 8 つの extended barrel モジュールの信号を受信するので、A-side と C-side 合わせて計 16 枚の TMDB がある。各 TMDB では測定されたトラックのエネルギーに閾値を設けることでトリガー判定が行われ、G-Link 光信号によって 3 台のエンドキャップセクター用 SL にその情報が送信される。

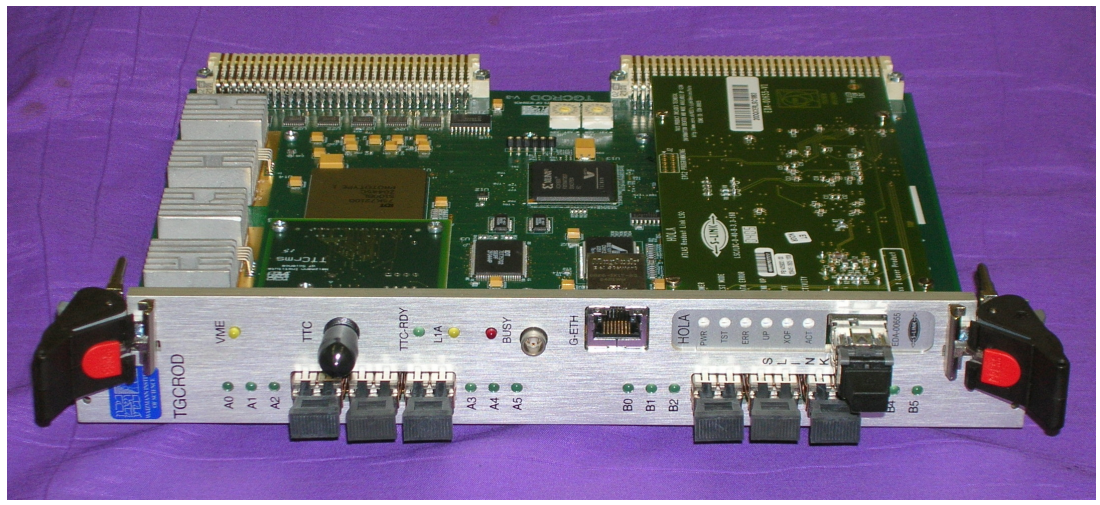


図 4.8　TGC ROD の写真 [22]。

4.1.4　TGC の読み出しエレクトロニクス

4.1.3 節で触れたように、PS Board および SL 上に搭載されている SLB ASIC はデータを読み出す機能も持っている。本小節では SLB 以降の TGC の読み出しエレクトロニクスについて説明する。

Star Switch (SSW)

PS Board 上の SLB から読み出されたデータは、TGC BW の端にあるラック内の VME クレートに挿入されている Star Switch (SSW) というボードに渡される。1 枚の SSW は最大で 23 枚の SLB ASIC からデータを受信できる[*7]。受信されたデータは RX FPGA にて圧縮 (ゼロサプレス) され、バッファされる。バッファされた各 SLB からのデータは TX FPGA によって 1 イベントごとまとめられ、G-Link 規格を用いた光信号によって後段の Readout Driver に送信される。なお、SL 上の SLB から読み出されたデータを処理する SSW は USA15 に設置されていた。

SSW はデータの読み出しに加え、フロントエンド回路の制御も担う。4.1.5 節に後述する HSC モジュールは SSW が持つ Control FPGA のレジスタを経由して、PS Board や 4.1.6 節で説明する Service Patch Panel 上の TTCrx というチップを制御できるようになっている。

Readout Driver (ROD)

SSW から来る読み出しデータはすべて USA15 にある TGC の Readout Driver (ROD) に送られる。図 4.8 がその写真である。ROD は SSW から受け取った読み出しデータを L1ID ごとにまとめ、ヘッダーやトレイラー情報を付けることで、後段の ROS (3.2.6 節参照) が受信できるような形にデータを加工する役割を持つ。1 枚の ROD は 1 つの 1/12 セクターを担当しており、S-LINK というプロトコルを用いて ROS にデータを送信する。加えて、ROD は読み出しパスの処理が間に合っていないとき (例えば SSW の RX FIFO がオーバーフローを起こしたとき)、TGC の TTC システムを経由して CTP に busy 信号を送ることで、L1A の出力を一時的に止めることができる。

[*7] PS Board には 2 枚または 3 枚の SLB が搭載されている。SSW は 2 枚の SLB を持つ PS Board 7 台分、3 枚の SLB を持つ PS Board 3 台分のデータを受け取れる。すなわち、最大で $2 \times 7 + 3 \times 3 = 23$ 枚の SLB からデータを受信できる。

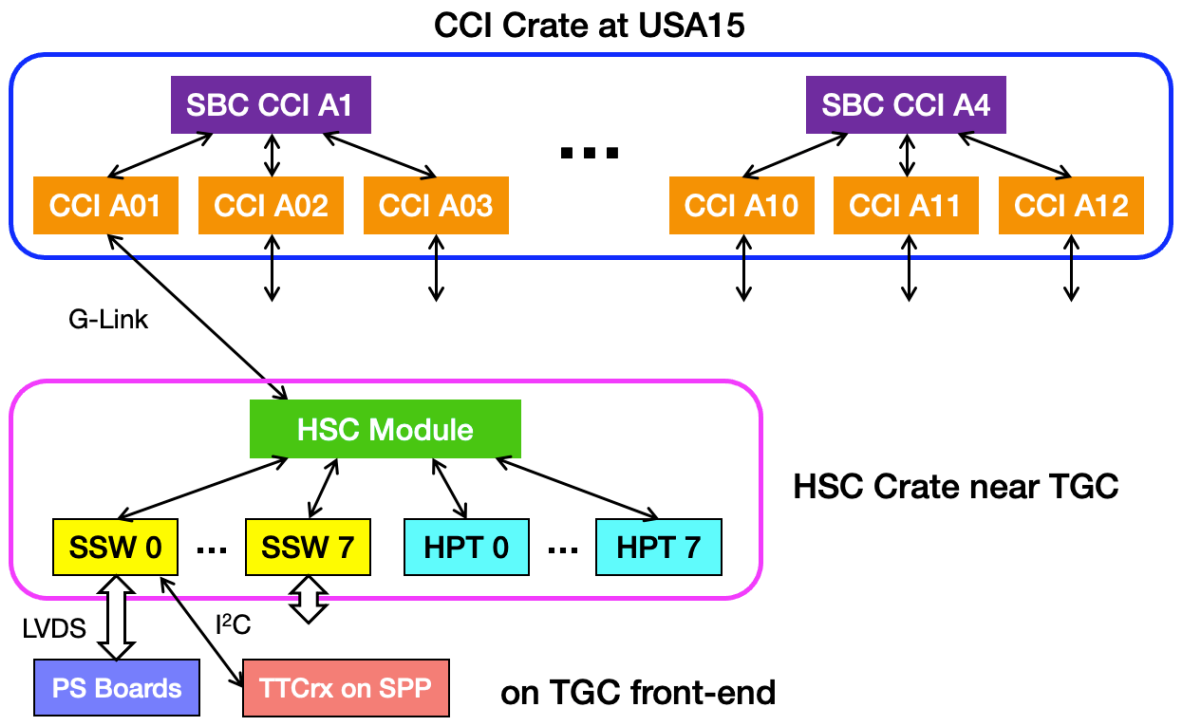


図 4.9　TGC のフロントエンド回路制御システムの概要。ここでは A-side の TGC BW を制御するためのモジュールのみを記した。実際は TGC EI/FI 用の制御するための CCI などもある。

4.1.5　TGC のフロントエンド回路制御システム

UX15 (ATLAS 実験室) にある PS Board や HPT ボードなどのフロントエンドエレクトロニクスを遠隔で制御するために、TGC システム独自のフロントエンド回路制御用エレクトロニクスが用いられている。図 4.9 にそれらを用いた回路制御パスの概要を示す。この小節ではこれらの電気回路制御エレクトロニクスについてまとめる。

Control/Config Interface (CCI) モジュール

Control/Config Interface (CCI) モジュールは USA15 に設置されている、フロントエンドエレクトロニクスを遠隔で操作するための回路制御ハードウェアである。1 台の CCI モジュールは TGC の 1/12 セクターを 1 つ担当するので、TGC BW を制御するための CCI モジュールは A-side と C-side で合計 24 個ある。CCI モジュールは同じ VME クレート内の Single Board Computer (SBC) を用いてネットワーク経由で制御できるようになっている。各 CCI モジュールは G-Link 規格の光信号を用いて、UX15 の TGC 検出器の脇に位置する HSC クレート内にある HSC モジュールと通信を行う。

HPT/SSW Controller (HSC) モジュール

HPT と SSW は TGC 検出器の近くに位置する HPT/SSW Controller (HSC) クレート内に入っており、HSC モジュールによって VME 経由で制御される。HSC クレートは TGC の 1/12 セクターについて 1 台あり、それぞれに 8 枚の HPT ボードと 8 枚の SSW が挿入されている。HSC モジュールは HSC クレートの VME バスのマスターとして機能し、CCI から受け取った制御信号を HPT や SSW に配る。さらに SSW は、LVDS シリアル通信によって PS Board を制御する機能や、I^2C 通信 (5.5.1 節参照) によって PS Board に TTC 信号を配布する Service Patch Panel 上の TTCrx というチップを制御する機能を持っている。

TGC のフロントエンドシステムでは、このような形で CCI モジュールや HSC モジュール、さらには SSW を用いて制御信号を伝播させることで、遠隔でフロントエンドエレクトロニクスの制御を実現している。

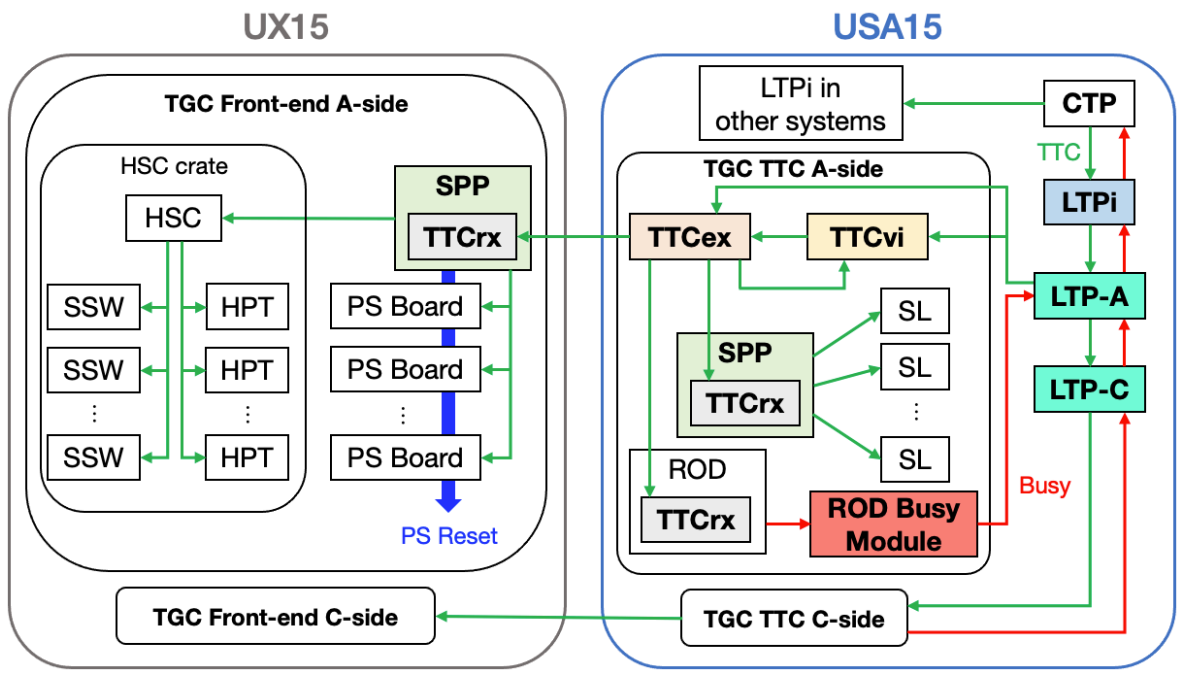


図 4.10　Level-1 ミューオンエンドキャップトリガーおよび TGC の TTC システムの概要。USA15 は ATLAS 回路室、UX15 は ATLAS 実験室である。緑の矢印は TTC 信号全般を、赤い矢印は busy 信号を、青い矢印は PS Reset 信号を表している。なお A-side と C-side は対称的な構造を持つので、C-side の詳細は省略してある。

4.1.6　TGC の TTC システム

CTP から配布される TTC 信号は各検出器の TTC システムによって適切なモジュールに配布される必要がある。Level-1 ミューオンエンドキャップトリガーおよび TGC の TTC システムの概要を図 4.10 に示す。本小節では TGC の TTC システムで用いられているエレクトロニクスについてまとめる。

Local Trigger Processor Interface (LTPi)

Local Trigger Processor Interface (LTPi) は CTP と後述する Local Trigger Processor (LTP) の間を結び、どのクロックを用いるかを制御するモジュールである。実際の実験ではもちろん CTP のクロックを用いるが、ATLAS の一部分だけで試験を行いたい場合は LTP のボード上の発振器など別のクロックソースを用いることがあるため、このようなモジュールが用意されている。また LTPi を用いることで CTP を介さずにとある LTP をマスタークロックとして複数のシステムを走らせることもできる [23]。

Local Trigger Processor (LTP)

Local Trigger Processor (LTP) は各サブシステムにおいて TTC 信号を司るモジュールである。TGC の TTC システムでは、CTP から LTPi を経た TTC 信号はまず A-side 用の LTP (図 4.10 の LTP-A) にまず入る。この TTC 信号は後段の A-side の TTC システムおよび C-side 用の LTP (LTP-C) に渡される。LTP-C は C-side の TTC システムに TTC 信号を配布する。また、TGC の単独試験 (standalone test) を行う場合は LTP-A がマスターとなり、TGC システム全体の TTC 信号の出力を担うことになる。

TTC VMEbus Interface (TTCvi)

TTC VMEbus Interface (TTCvi) は LTP から Orbit 信号、L1A、ECR を受信する。BC クロックのみ LTP からではなく後述の TTCex から受信するようになっている。TTCvi は受信した Orbit 信号から BCR を生成する役割を担っており、A チャンネルを使って L1A を、B チャンネルを使って BCR や ECR などの信号を TTCex に送信する [24]。また TTC を Test Pulse モードに設定することで、TGC 独自の user defined な信号として、Orbit 信号

と同期した Test Pulse Trigger 信号を生成することができる。ASD Test Pulse や、SLB ASIC から打つ SLB Test Pulse はこの Test Pulse Trigger 信号と同期して生成される。

TTC Encoder/Transmitter (TTCex)

TTC Encoder/Transmitter (TTCex) は TTCvi の A および B チャンネルから来る TTC 信号を光信号に変換し、最大で 10 個の出力を行えるファンアウトモジュールである。動作には BC クロックが必要なため、LTP から直接 BC クロックを受け取る。TGC の TTC システムにおける TTCex は UX15 (ATLAS 実験室) にある Service Patch Panel (SPP) や USA15 にある SPP や ROD に TTC 信号を配る。

TTCrx

TTCrx は TTC 信号を受信し、その信号を遅延させる役割を持つ IC チップである。LHC と同期して制御されねばならない各モジュールには TTCrx を搭載してそれらに適切な遅延をかけることで、粒子の ToF やケーブル長を考慮したタイミング調節をシステム全体で行うことができる。TTCrx 以降のケーブルはすべて等長配線されねばならない。TGC の TTC システムでは SPP や ROD に TTCrx が搭載されている。

Service Patch Panel (SPP)

Service Patch Panel (SPP) は TTCex から送信される TTC 信号を受信し、それを配布するための TGC システム固有のモジュールである。SPP は TTCrx を用いて TTC 信号を受信し、タイミング調節のために適切な遅延をかけられるようになっている。フロントエンドにある SPP は各 PS Board および HSC モジュールに TTC 信号を配る。PS Board は自身に搭載されている PP ASIC および SLB ASIC に TTC 信号を渡す。HSC モジュールは VME 経由で SSW と HPT に TTC 信号を渡す。また SPP は PP ASIC と SLB ASIC をリセットするための TGC 独自の user defined な TTC 信号である、PS Reset 信号を PS Board に送る役割も担っている。USA15 にある SPP は SL に TTC 信号を配布する。

ROD Busy Module

ROD Busy Module は各サブシステムの busy 信号をまとめ、LTP にその情報を送る役割を持つ。TGC の TTC システムにおける ROD Busy Module は各サイドに 12 台ある ROD の busy 信号をまとめ、LTP-A/C にその busy 信号を送る。使用していない ROD と繋がっている busy 線をマスクする機能も持っている。

Delay Module

図 4.10 には掲載していないが、TGC の TTC システムでは Delay Module というボードを用いて LTP、TTCvi、TTCex などの間の TTC 信号に適切な遅延をかけている。また、Delay Module は TTCrx のリセット信号をブロードキャストコマンドとして TGC の TTC システム全体に送る機能も持っている。A-side と C-side の TTC システムのそれぞれには 3 台の Delay Module がある。

4.2 Level-1 ミューオンエンドキャップトリガーの Run 3 アップグレード

2022 年に開始予定となった LHC Run 3 では、最高瞬間ルミノシティは Run 2 と同じに制限されるが、その高い輝度をより長い時間保つことで Run 2 より効率的に物理データを収集する。また重心系エネルギーを 13 TeV から 13.5 – 14 TeV に引き上げる可能性がある。ATLAS 実験では、p_{T} が数十 GeV のレプトンが出てくる物理現象に対するアクセプタンスを維持するために、シングルレプトントリガーの p_{T} 閾値を Run 2 のときと同じに保たねばならない。このため、Run 3 に向けて Level-1 Trigger を中心としたアップグレード作業が 2021 年 1 月現在も進行し

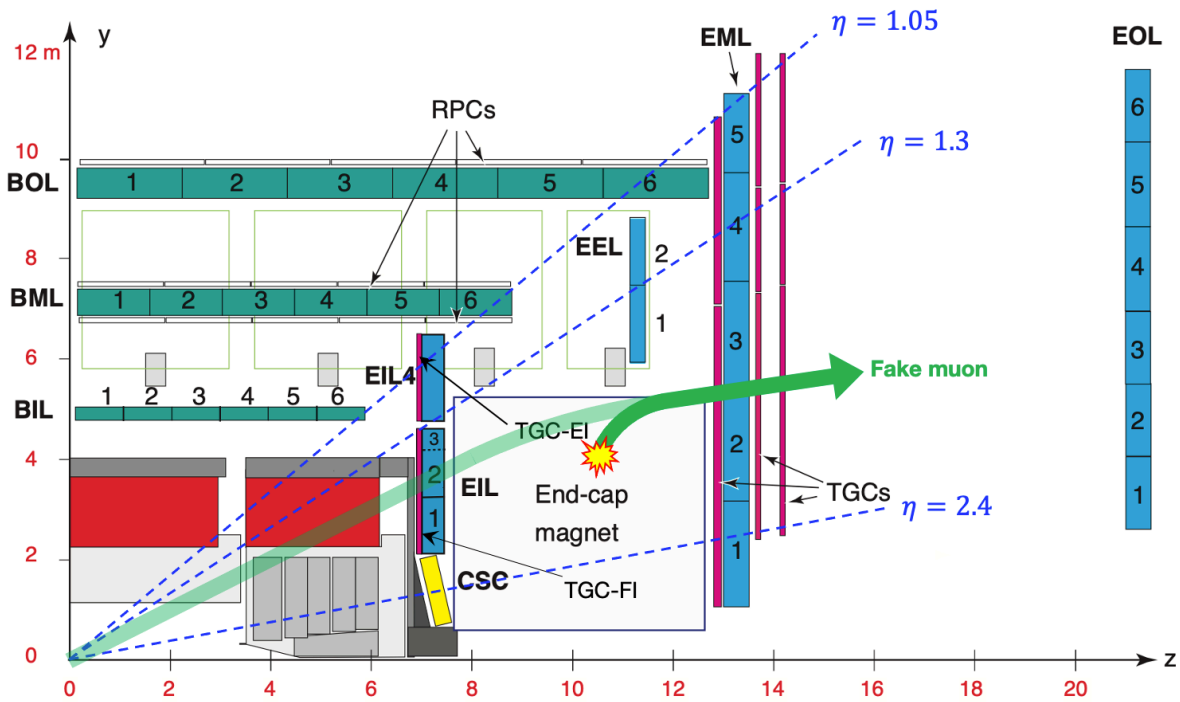


図 4.11　Level-1 ミューオンエンドキャップトリガーにおけるフェイクトリガーの模式図 [14]。緑の矢印は endcap toroid magnet から生じた粒子の飛跡を表しており、これはフェイクトリガーを生じさせうる。

ている。

Level-1 ミューオンエンドキャップトリガーでは、陽子陽子衝突に由来するミューオン以外の荷電粒子が原因となって出力されるフェイクトリガー信号の削減がこれまでの課題となってきた。その主な要因は図 4.11 に示すように、endcap toroid magnet や beam shielding から生じる、低い p_{T} の陽子であることが知られている。これらの陽子が高い p_{T} を持つミューオンと同じ角度で TGC に入射した場合、フェイクトリガーが出力されてしまう。2012 年に取得したデータの解析結果によると、Level-1 ミューオンエンドキャップトリガーによって出力されたトリガーの約 90% がフェイクであった [25]。その後の Run 2 では Tile Calorimeter D 層および TGC EI/FI を Inner Coincidence に導入することで、$1.05 < |\eta| < 1.92$ の領域におけるフェイクトリガーの削減に成功している。

しかし $1.92 < |\eta| < 2.4$ の領域では Inner Coincidence を行うための検出器がないためまだフェイクトリガーが多く、$1.05 < |\eta| < 1.92$ においてもまだフェイクを削減する余地がある。すなわち Run 3 ではこれらのフェイクトリガーを削減するべく、磁場領域より内側に New Small Wheel および RPC BIS78 という 2 つの新しい検出器を導入する。図 4.12 に示すように、これらの検出器を Level-1 ミューオンエンドキャップトリガーのトリガーロジックに組み込むことで、フェイクトリガーの 90% 以上が削減できると見積もられている [26]。加えて新しい検出器の導入にしたがって、Level-1 ミューオンエンドキャップトリガーのトリガーロジックおよびエレクトロニクスも大幅に刷新される。本節ではこれらの新しい検出器およびエレクトロニクスについて説明する。

4.2.1　New Small Wheel (NSW)

Run 3 から新たに導入される New Small Wheel (NSW) は $1.3 < |\eta| < 2.7$ の領域を覆う、円盤状のガス検出器である。NSW は図 4.13(a) にあるように ϕ 方向に、8 つの large sector と 8 つの small sector に分割されている。また、NSW は Run 2 で使われていた Small Wheel と入れ替わる形で設置されるので、現行の Small Wheel に含まれる TGC FI、MDT の EIL チェンバー、および CSC はなくなる。すなわち NSW は、TGC FI が担っていたミューオントリガー検出器の役割および MDT と CSC が行なっていたミューオン飛跡の精密測定の役割の両方を果たさねばならない。そのため図 4.13(b) に示すように、NSW は主にトリガー検出器として機能する small-strip TGC (sTGC) と主に飛跡の精密測定を行う Micromegas (MM) という 2 種類の検出器から構成されている。4.2.3 節に後述するように、これらの検出器の信号を用いてオンラインで飛跡を再構成し、その飛跡の情報をトリガーに用

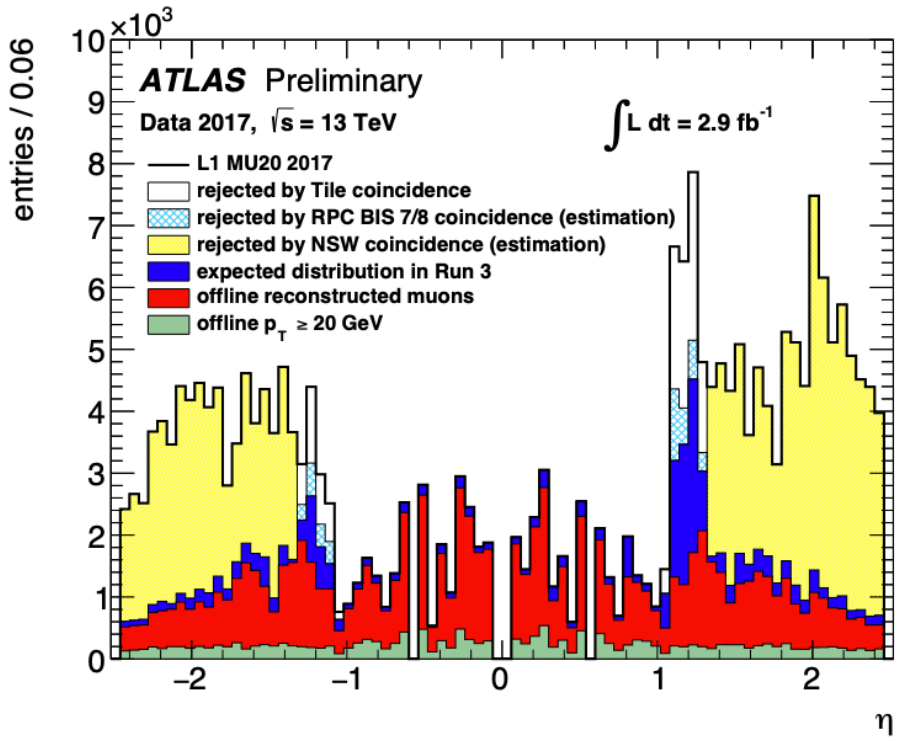
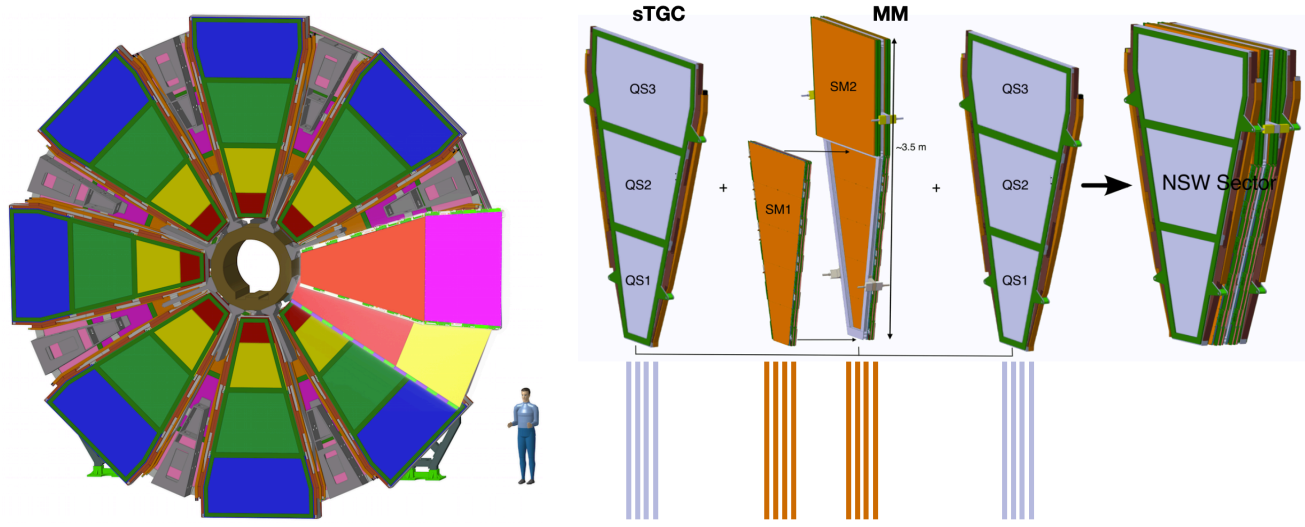


図 4.12　Run 3 における Level-1 ミューオントリガーを用いて選出したミューオン候補数の η 分布の予測 [27]。2017 年に L1_MU20 ($p_{\mathrm{T}} \geq 20$ GeV のミューオンを Level-1 Trigger で要求するトリガーメニュー) を用いて取得した積分ルミノシティ 2.9 fb^{-1} のデータを使っている。Tile Calorimeter とのコインシデンスによって実際に黒枠の白い領域の数だけミューオン候補が削減された。また NSW および RPC BIS78 の導入によって、それぞれ黄色の領域および水色の斜線領域の数だけミューオン候補が削減されると見積もられている。結果的にフェイクトリガーは 90% 減り、最終的なトリガーレートは 45% 程度減ると予測されている。



(a) NSW の全体図　　(b) NSW の small sector の構造

図 4.13　NSW の概要 [28]。図 4.13(a) がその全体図である。外側がマゼンタ色の部分が large sector、外側が黄色の部分が small sector である。図 4.13(b) は small sector における層の構造を表す模式図である。各 4 層の 2 つの sTGC の間に 4 層 ×2 の MM が挟まれる構造になっている。Large sector も同様の構造を持つ。

いる。NSW を導入することで、Inner Coincidence のカバー領域が $|\eta| < 2.4$ まで拡張され、さらに新たな角度情報を用いたトリガーロジックの導入によってフェイクトリガーを大幅に削減できるようになる。

Small-strip TGC (sTGC)

Small-strip TGC (sTGC) は NSW を構成する検出器の 1 つで、主にトリガー検出器の役割を果たす。sTGC の検出器の動作原理と構造は図 4.14 に示すように、TGC とほぼ同じである (3.2.5 節を参照)。しかし sTGC では R

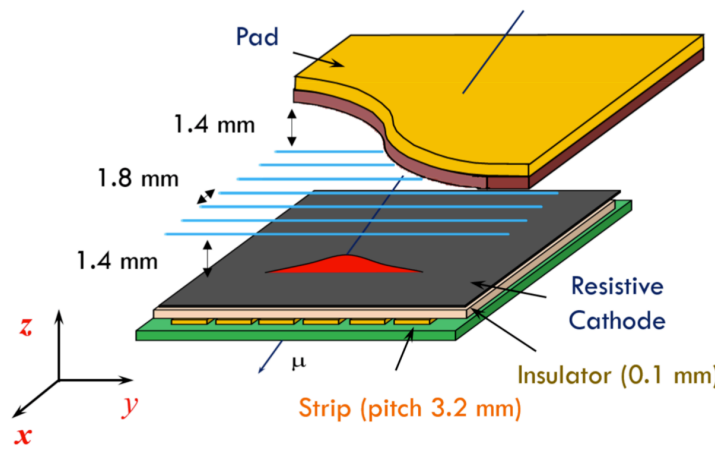


図 4.14　sTGC 検出器の構造 [28]。ワイヤーを用いて ϕ を測定し、ストリップを用いて η を測定する。η を精密に測定するためにストリップの読み出し間隔は TGC と比べて小さくなっている。ガス層を挟んでストリップと反対側にあるパッドでも電荷の読み出しを行う。

方向に伸びるワイヤーを用いて ϕ を測定し、ϕ 方向に走るストリップを用いて η を測定する。特に η を精密に測定するために、ストリップの読み出し間隔は 3.2 mm と、TGC より遥かに小さくなっている。これが small-strip TGC という名前の由来である。sTGC の位置分解能は粒子の入射角に依存して $60-150$ μm を達成できるとされている。また、ガス層を挟んでストリップとは反対側にあるカソード面には銅でできたパッド (pad) があり、こちらでも電荷の読み出しができるようになっている。パッドを使って sTGC の各層のコインシデンスを取ることで粒子の大まかなヒット位置を特定し、読み出すべきストリップを決める。このような段階的な計算を行うことで、精密な位置情報をトリガーに用いることができる。さらに、オフラインにおける飛跡再構成のためにすべてのワイヤー、ストリップ、およびパッドの電荷が読み出される。

Micromegas (MM)

Micromegas (Micro Mesh Gaseous Structure, MM) はワイヤーを用いないガス検出器で、NSW における飛跡の精密測定を担う。図 4.15 にその動作原理の模式図を示す。MM のドリフトカソード面と読み出し電極がある面の間の領域は混合比 93:7 の Ar/CO_2 によって満たされている。このガス層はステンレススチール製のメッシュによってドリフト領域と増幅領域に隔てられている。荷電粒子がガス層を通過すると、まずはドリフト領域にてガスが電離し、生じた電子は 600 V/cm の電場によってメッシュがある方向にドリフトする。メッシュと高抵抗ストリップの間の増幅領域における 40 kV/cm の高電場によって、ドリフトしてきた電子は電子雪崩を起こす。この電子雪崩に伴って陽イオンも生じるが、増幅領域の距離が短いため、それらの陽イオンがメッシュまで戻る時間は高々 100 ns である。そのため MM は通常のドリフト検出器よりも応答が速い。増幅された電子雪崩の電荷は、放電を防ぐために導入された高抵抗のストリップを経て、読み出しストリップに伝播される。MM の位置分解能は入射角が 40° 未満の粒子に対して 100 μm より小さいことが示されている[*8][25]。

4.2.2　RPC BIS78

図 3.11 や図 3.15 に示すように、large sector には TGC EI が設置されているが、small sector の同じ領域には barrel toroid magnet があるため、この endcap toroid magnet より内側の η 領域にはトリガー検出器がない。Run 3 ではこの問題を解決するべく、図 3.11(b) にある MDT の BIS チェンバーの 7 番および 8 番を、RPC と小さい MDT (small-diameter MDT, sMDT) を組み合わせた新しい BIS78 ステーションに置き換える。この新しい RPC を RPC BIS78 と呼ぶ。図 4.16 に新しい BIS78 ステーションの模式図を載せる。

RPC BIS78 は $1.0 < |\eta| < 1.3$ の領域を覆い、3 層のガス層を持つ。その 1 層の断面図を図 4.17 に示す。限られ

[*8] NSW に入射するミューオンの入射角の範囲は 8°–30° である。

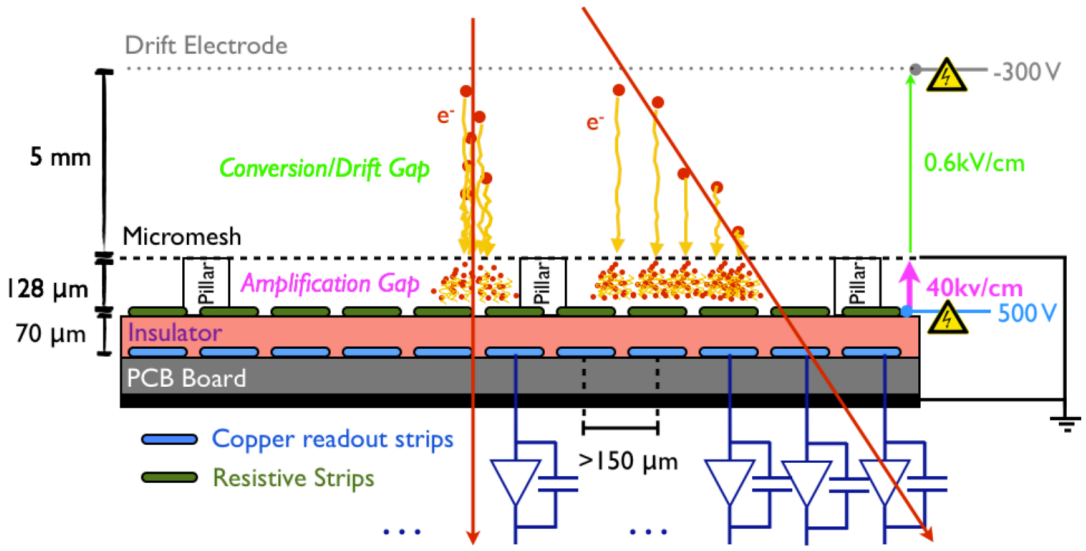


図 4.15　Micromegas の動作原理の模式図 [25]。ドリフトカソード面と電荷の読み出し電極がある面の間の領域は、金属のメッシュによってドリフト領域と増幅領域に隔てられている。NSW で使用される MM は放電を防ぐために高抵抗のストリップ (resistive strip) を導入している。

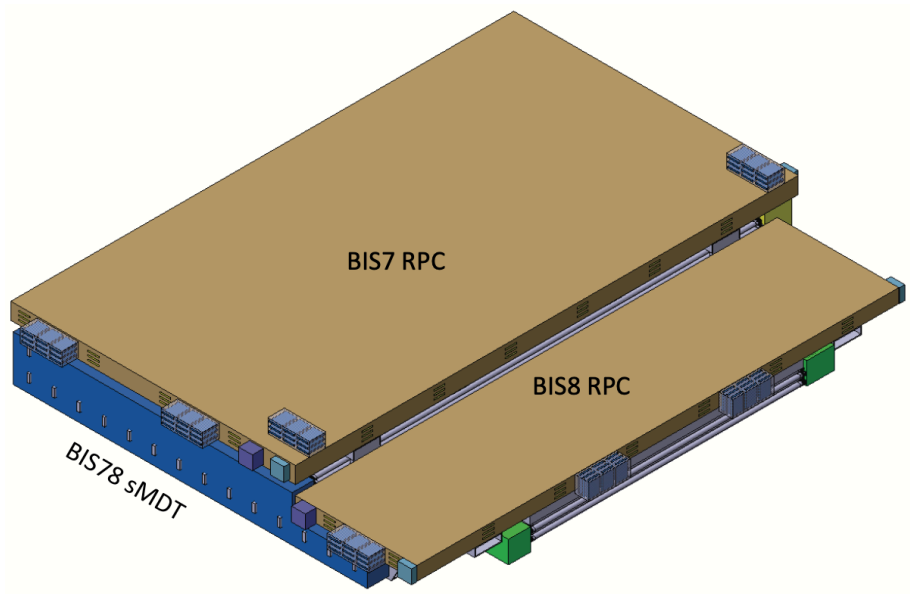


図 4.16　Run 3 から設置される BIS78 ステーション [29]。RPC BIS78 および sMDT BIS78 から構成される。ステーションの厚みは Run 2 まで使われてきた MDT の BIS78 チェンバーと同じである。

たスペースに設置せねばならないため、RPC BIS78 のガス層の厚みはおよそ 1 mm で、従来の RPC の半分になっている。このことによって、必要な印加電圧は 9.8 kV から 5.4 kV に減少し、時間分解能も 0.4 ns に改善している。また、ガス層は従来の RPC と同様に 2 次元方向に伸びるストリップに挟まれており、(η, ϕ) の測定がされる。位置分解能は 1 mm 程度である。

4.2.3　新しいトリガーエレクトロニクス

NSW と RPC BIS78 の導入によって、Level-1 ミューオンエンドキャップトリガーのトリガーロジックは一新される。これに伴い、多くのトリガーエレクトロニクスが新たに導入される。図 4.18 に Run 3 の Level-1 ミューオンエンドキャップトリガーで用いられるトリガーエレクトロニクスの全体像を示す。本小節ではこれらの代表的なエレクトロニクスについてまとめる。

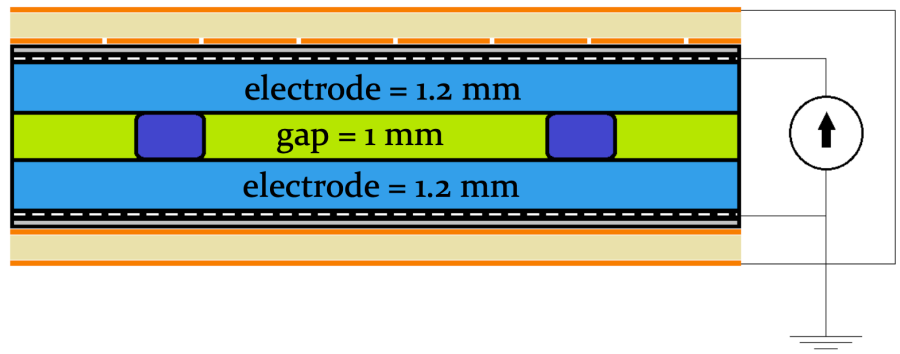


図 4.17　RPC BIS78 の 1 層の断面図 [30]。RPC BIS78 はこの層を 3 つ含む。

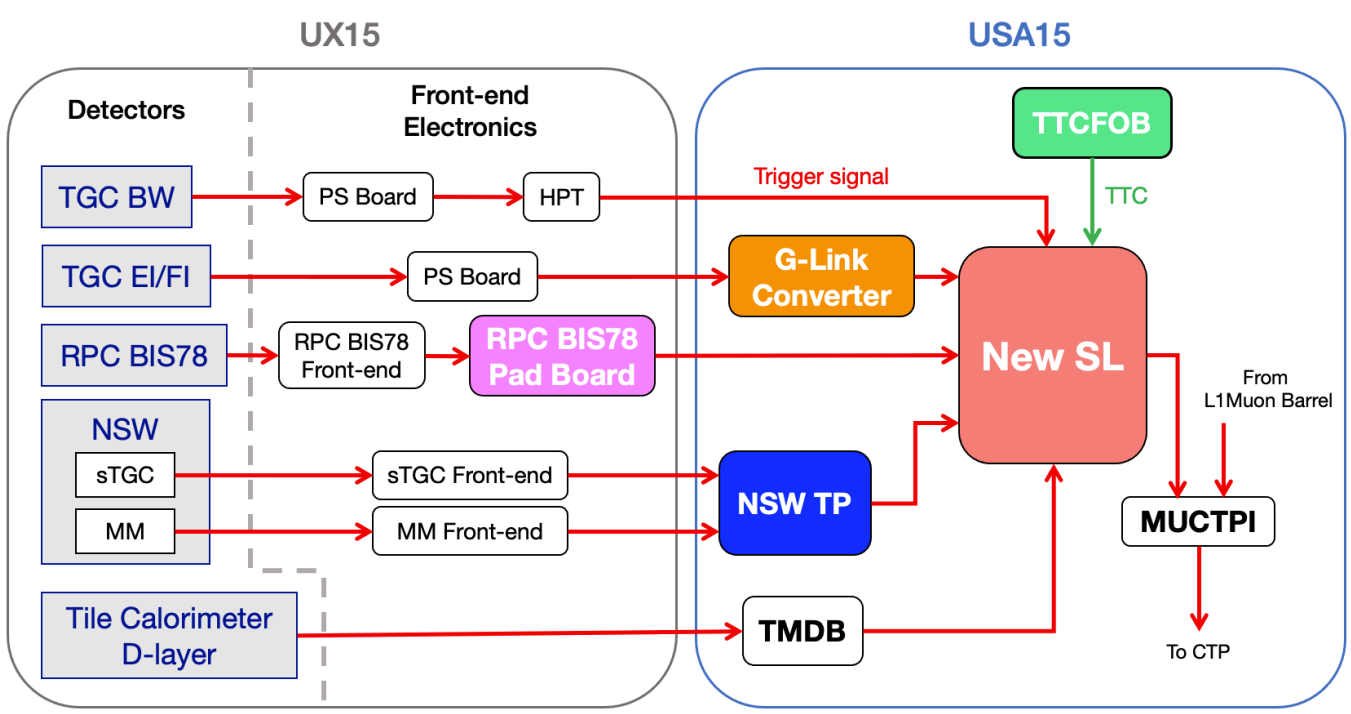


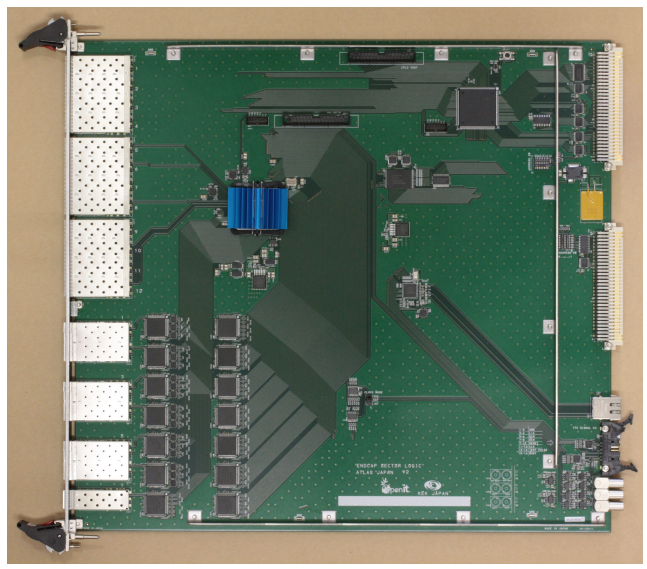
図 4.18　Run 3 の Level-1 ミューオンエンドキャップトリガーシステムの全体像。簡単のため、一部エレクトロニクスは省略してある。メインのトリガー判定ボードである New Sector Logic (New SL) は TGC BW、TGC EI/FI、RPC BIS78、NSW、および Tile Calorimeter からのヒット情報を基に、最終的なトリガー判定を下す。4.2.3 節ではこの図で色がついている 5 つの新規エレクトロニクスについて説明する。

NSW Trigger Processor (NSW TP)

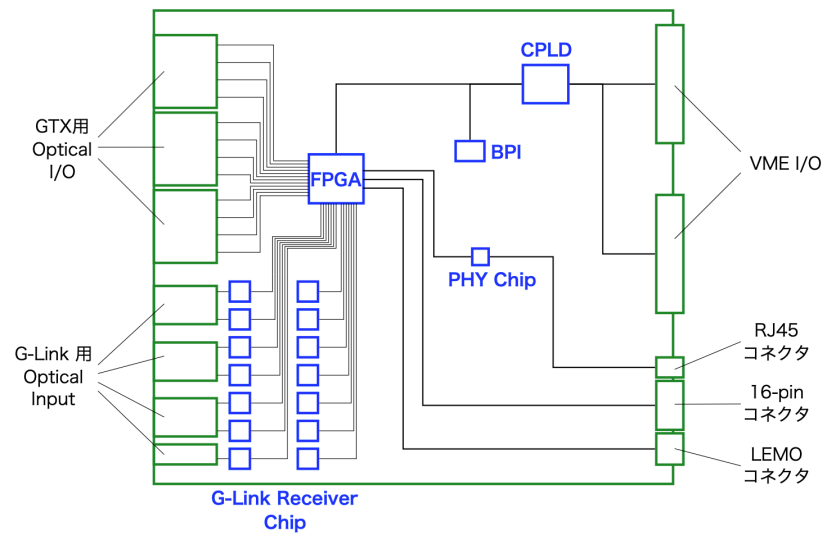
sTGC と MM のトリガーに関するヒット情報はフロントエンド回路では別々に処理され、USA15 に設置される予定の NSW Trigger Processor (NSW TP) というハードウェアによって組み合わされる。NSW TP は再構成した飛跡のトラック情報を 8b/10b 方式のシリアル通信規格の光信号によって後段の New Sector Logic に送る。トラック情報には η と ϕ の位置情報に加えて、飛跡の角度情報 $\mathrm{d}\theta$ が含まれる。

RPC BIS78 Pad Board

RPC BIS78 Pad Board は UX15 に設置される予定のフロントエンドボードで、RPC BIS78 の 3 層のヒット情報を用いて 2/3 コインシデンスを取る。再構成した飛跡のトラック情報は 8b/10b 方式の光通信によって後段の New Sector Logic に送られる。トラック情報には 2 次元位置 (η, ϕ) と飛跡の角度情報 $(\mathrm{d}\eta, \mathrm{d}\phi)$ が含まれる。Pad Board は各サイド 16 個ある BIS78 ステーションのそれぞれに 1 枚あるが、後段のエンドキャップセクター用の New Sector Logic は 24 台あり、これらは 1 対 1 の対応関係にない。そのため Pad Board から来る光信号は USA15 にある Optical Splitter によって複数の New Sector Logic に分配される。



(a) New SL の写真



(b) New SL の主な機能

図 4.19　New SL の主な機能。図 4.19(a) は New SL の写真である。図 4.19(b) は図 4.19(a) 上の主なチップと配線を示した模式図である [19]。緑色のブロックは I/O を、青色のブロックはチップを示している。

New Sector Logic (New SL)

Run 3 の Level-1 ミューオンエンドキャップトリガーでは、NSW や RPC BIS78 の導入によってトリガー判定に使用せねばならない情報が大幅に増える。そのため、Level-1 ミューオンエンドキャップトリガーの最終的なトリガー判定を行うバックエンドボードである Sector Logic も新たに開発された。この新しい Sector Logic を New Sector Logic (NewSL) と呼ぶ。各 New SL は 4.1.3 節で述べた Run 2 の SL と同様に、2 つのトリガーセクターを担当する。すなわち A-side と C-side のそれぞれに 24 台のエンドキャップセクター用の New SL と 12 台のフォワードセクター用の New SL がある。

図 4.19 に沿って、New SL の各機能についてまとめる。NSW、RPC BIS78、および後述の G-Link Converter を経た TGC EI/FI からの光信号の受信は SFP+ 光トランシーバにて行われる。SFP+ は光信号と電気信号の間の変換を行うドライバーで、FPGA に搭載されているトランシーバのインターフェースとなる。また TGC BW と TMDB からの光信号の受信は G-Link 用の SFP 光トランシーバにて行われる。SFP 光トランシーバによって受信されたシリアル信号はまず G-Link 受信チップにて LHC のクロック信号と同期してパラレル信号に変換される。図 4.20 に FPGA ファームウェアの全体像を示す。FPGA ではこれらのすべての信号を用いてコインシデンスが取られ、LUT を用いた p_{T} 計算が行われる。トリガー判定の情報は GTX トランシーバと SFP+ 光トランシーバを経て光信号にて後段の MUCTPI に送信される。

New SL は SiTCP という FPGA と Ethernet を接続する技術を用いて、TCP/IP 通信にてトリガーデータを読み出しシステムに送る。Run 2 で使用された SL では SLB ASIC を用いてデータをバッファしていたが、New SL では FPGA がその機能を果たす。後述の TTC Fanout Board から 16-pin のフラットケーブルにて TTC 信号を受け取るが、L1A 信号が来た場合は FPGA のファームウェアに実装されている L1 buffer にため込んでいる検出器ヒット情報とトリガー判定情報を読み出し、Derandomizer にて 16 ビットずつに分割する。これらの分割されたデータはゼロサプレスロジックで圧縮され、Ethernet PHY チップに送られる。最後に、これらのデータは RJ45 コネクタに繋がっている Ethernet ケーブルを通じて、TCP/IP 通信にて後段の読み出しシステムに送られる。

VME バスを用いた制御のために、New SL に CPLD[*9]を搭載している。CPLD を用いることで、FPGA への回路デザインの書き込みや FPGA との通信を VME 経由で行える。また BPI[*10]に FPGA の回路デザインを保存して

[*9] Complex Programmable Logic Device の略。FPGA とは異なって不揮発性メモリを有するので、電源を落としても電源を落とす前のデータが残る。

[*10] Byte Peripheral Interface の略。

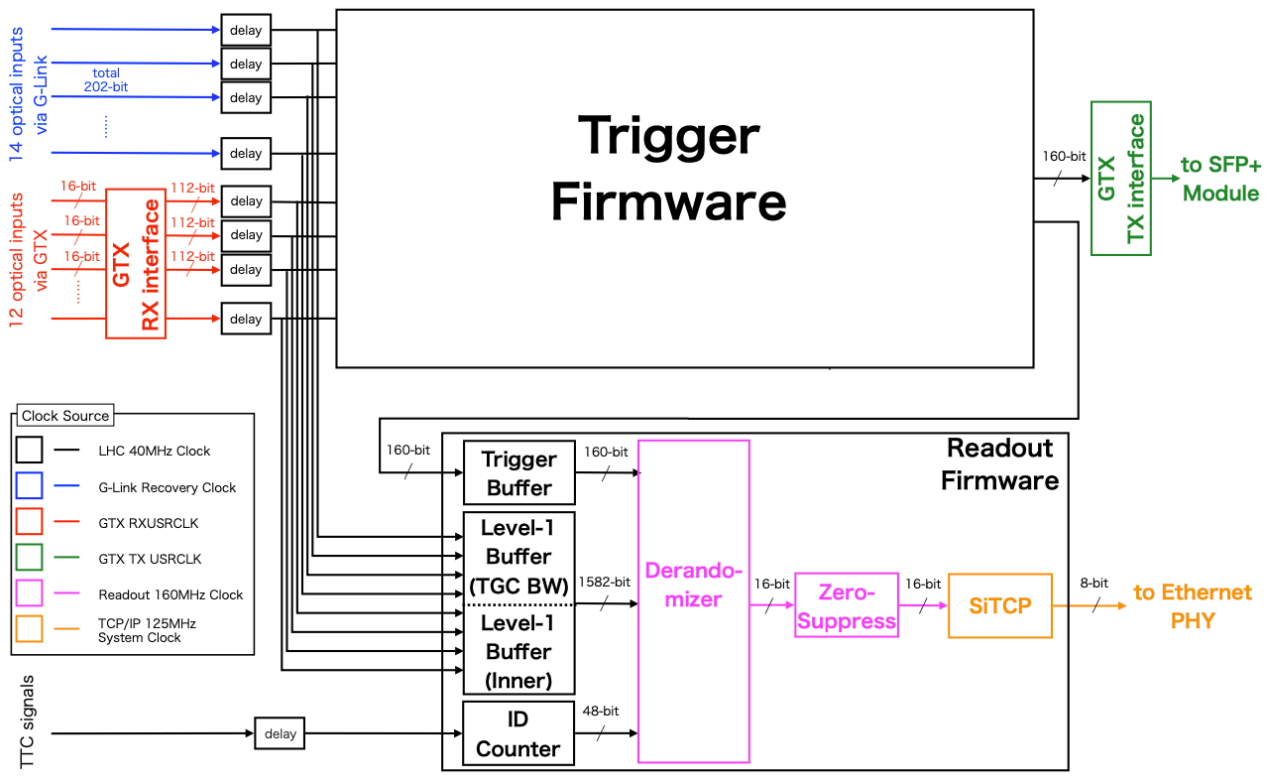


図 4.20　New SL の FPGA ファームウェアの全体像 [31]。上がトリガーパス、下が読み出しパスである。トリガーパスでは 8b/10b 規格および G-Link 規格で受信した検出器ヒット情報を用いてトリガー判定を行い、8b/10b 規格でその情報を出力する。読み出しパスでは検出器ヒット情報とトリガー判定情報を L1 buffer にためておき、L1A が来たらそれらの情報を Ethernet PHY チップに送信する。

おくことで、電源投入時に CPLD がその回路デザインを自動で書き込むようにできる。BPI の制御も CPLD を介して行える。

G-Link Converter

TGC EI/FI のトリガー信号は UX15 にある PS Board から G-Link 規格の光信号で送られてくる。NSW の検出器開発の遅れによって TGC FI が Run 3 でも使われる可能性に留意し、New SL は TGC EI/FI の信号を 8b/10b 規格の光信号で受信することになった。そのため TGC EI/FI の G-Link 信号をまとめ、8b/10b 信号に変換する役割を担う G-Link Converter が USA15 に導入された。G-Link Converter は New SL と全く同じハードウェアを用いるが、使用するファームウェアおよびソフトウェアが異なる。

図 4.21 に TGC EI/FI のチェンバーと G-Link Converter の関係を示す。各サイドの TGC EI/FI はそれぞれ 2 台の G-Link Converter によって担当されている。G-Link Converter は隣接する 3 つのセクターの情報をまとめ、1 台のエンドキャップセクター用の New SL にその情報を送信する。すなわち、2 台の G-Link Converter が担当する領域の境界にある 4 つのセクターの PS Board の信号は両方の G-Link Converter に配布される。つまり、1 台の G-Link Converter は 14 本の光ファイバーから G-Link 信号を受け、FPGA に搭載されている GTX トランシーバを用いて、12 台の New SL に 8b/10b 規格のシリアル信号を送信する。

TTC Fanout Board (TTCFOB)

TTC Fanout Board (TTCFOB) は TTC 信号を New SL に配布するために新たに開発されたバックエンドボードである。TTCrq というメザニンカードで、4.1.6 節で述べた TGC の TTC システムから TTC 信号を受信する。受信した TTC 信号は TTCrq 上の TTCrx チップを用いて適切に遅延をかけることができる。この TTC 信号は 12 台の New SL に、等長の 16-pin のフラットケーブルを通じて配布される。New SL は A-side と C-side を合わせて 72 台あるので、TTCFOB は計 6 台ある。

TTCFOB は New SL と同じく、VME を用いて CPLD 経由で FPGA と通信できる。TTCFOB の FPGA は

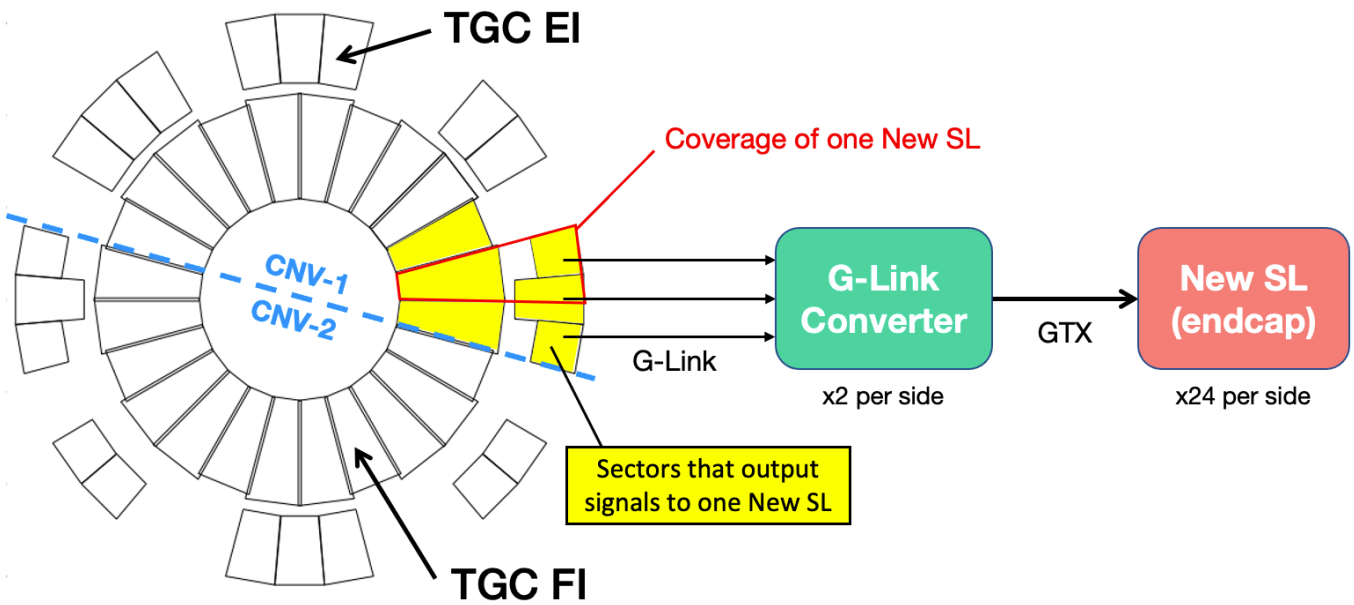


図 4.21　TGC EI/FI のチェンバーと G-Link Converter の関係。水色の点線が 2 台の G-Link Converter が担当する領域の境界を示している。また赤枠で囲まれている領域を担当する New SL は、黄色に塗られている領域の信号を G-Link Converter から受け取る。

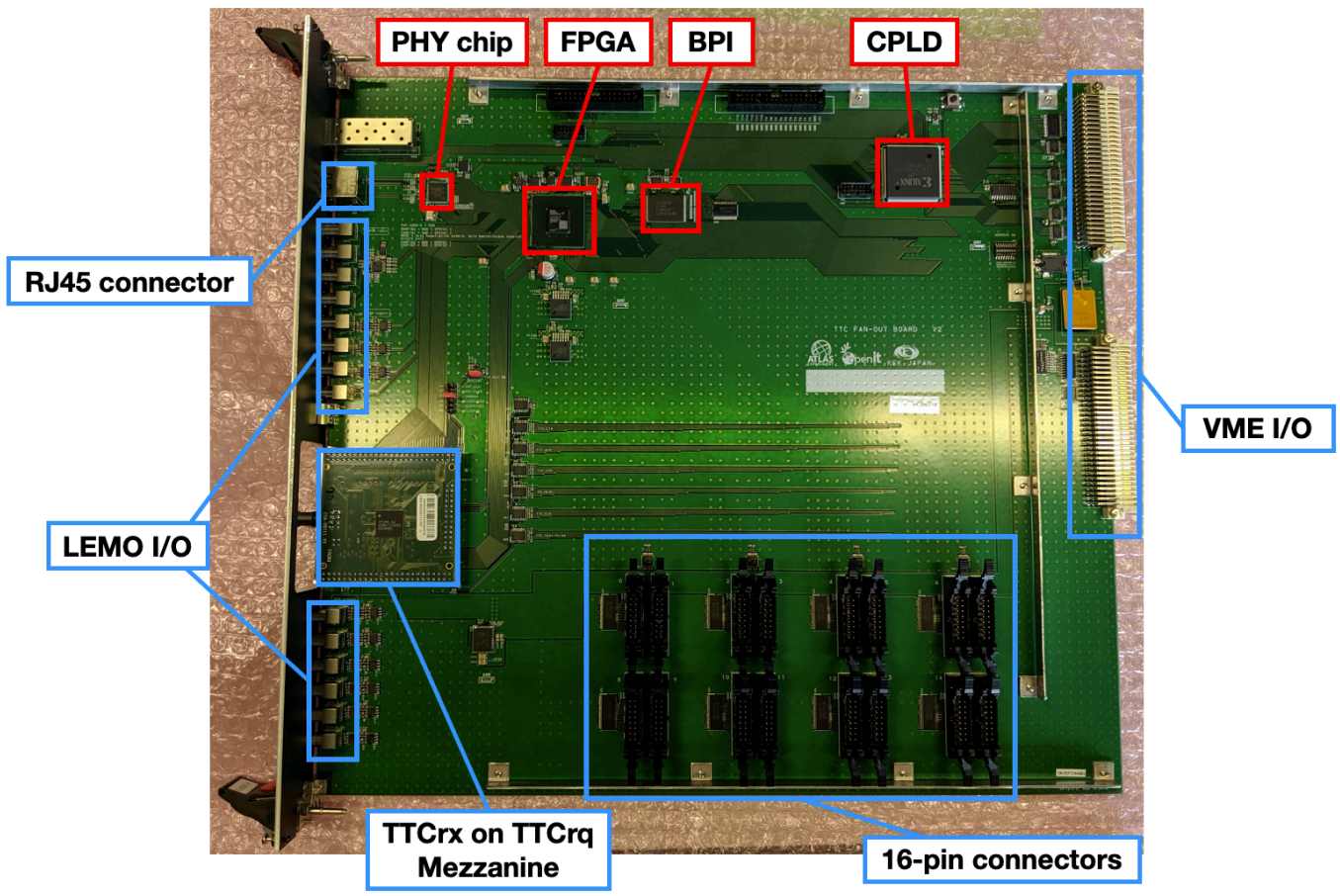


図 4.22　TTCFOB の主なチップと I/O。

TTCrx のレジスタ操作や TTC 情報の読み出し処理を担っている。前者については本研究の一環として通信手順を確立したので、第 5 章でより詳しく述べる。後者については、TTCFOB も New SL と同様に SiTCP 技術を用いて、L1A が来た際の BCID などの TTC 情報を TCP/IP 通信にて後段の読み出しシステムに送る。そのために PHY チップや RJ45 コネクタが実装されている。さらに、TTCFOB には複数の LEMO コネクタがある。図 4.22 の TTCrq メザニンより上側にある LEMO コネクタはユーザーが定義できるものであり、入力用と出力用がそれぞれ 4 つずつある。入出力はともに 3 つが NIM 規格で、1 つがオープンドレインになっている。これらは主に busy 信号の伝達に使用されている。詳しい配線については第 6 章で説明する。TTCrq メザニンより下側にある LEMO コネクタは TTCrq が受信した信号を出力するためのものであり、すべて NIM 規格である。上から順に L1A、BCR、ECR、PS RESET、BC クロック、Test Pulse Trigger 信号が出力できるようになっている。

4.2.4 トリガーデータ読み出しシステムのアップグレード

4.1.3 節の終盤で説明したように、Run 2 では SL 上の FPGA で処理されたトリガー判定情報は同じく SL 上にあった SLB ASIC を介し、SSW へと送られていた。SSW はマルチプレクサとして機能し、データを圧縮して後段の ROD にトリガーデータを送信していた。ROD はそれらのデータが正しいフォーマットに加工し、S-LINK 規格の光通信によって ROS にその加工データを送っていた。しかし Run 3 においては、NSW や RPC BIS78 の導入によって読み出さねばならないデータが大幅に増える。Run 2 で使用していた読み出しシステムはハードウェアベースであることから機能の拡張が容易ではないため、Run 3 では新たなトリガーデータ読み出しシステムを開発する必要がある。

そこで Run 3 では、FPGA やソフトウェアをベースとした、機能拡張性に富んだトリガーデータ読み出しシステムを新たに用いる。図 4.23 にアップグレード前後のトリガーデータ読み出しシステムの全体像を示す。Run 2 で SLB ASIC が担っていたバッファ機能はすべて New SL の FPGA 上に実装される。4.2.3 節でも述べたように、New SL は 100 kHz の頻度で L1A を受信し、バッファしていたトリガーデータを SiTCP [32] という TCP/IP 通信で送信する。複数台の New SL から送られてきたデータは市販のネットワークスイッチによって 1 本の線にまとめられ、後段の Software-based Readout Driver (SROD) という読み出しソフトウェアが走る市販の PC (SROD PC) に送信される。SROD PC では SROD ソフトウェアを用いて受信したデータを L1ID ごとにまとめて、ATLAS で決められたフォーマットに則ってイベントデータを作り、S-LINK 通信にてそのイベントデータを ROS に送る。なお本研究では SROD を中心としたこの高速読み出しシステムを ATLAS の DAQ 系に組み込むことで、Run 3 における Level-1 ミューオンエンドキャップトリガーのトリガーデータ読み出しパスを確立した。以下で Run 3 から新たに導入するハードウェアについて説明する。

SROD ソフトウェアを走らせるための高機能計算機サーバー (SROD PC)

Run 3 では 3 台の SROD PC で片サイド (A/C) のトリガーデータ、つまりは 1 台で TGC の 1/12 セクター 4 つ分 (より正確には 5.2.2 節に後述する New SL クレート 1 つ分) のトリガーデータを処理する。よって 2020 年 3 月に ATLAS 回路室である USA15 に、SROD ソフトウェアを走らせるための商業用の PC が 6 台導入された。今回導入された PC は Supermicro 社の SuperServer 5019P-WTR という 1U[*11]サイズのラックマウント型サーバーである [33]。この PC の主な仕様を表 4.1 に示す。ネットワークスイッチから来る New SL と TTCFOB のデータは 1 つの 10 GbE ポートで受信される。もう 1 つの 10 GbE ポートは IPMI[*12]を用いたモニタリングや制御に用いられる。また後述する PCIe S-LINK Card は PCIe 3.0 x16 スロットの 1 つに挿入されている。

[*11] U とはユニットの略で、サーバーの高さを表す単位である。1U = 1.75 インチ である。
[*12] Intelligent Platform Management Interface の略。OS を介さずにシステムを管理するためのインターフェース。

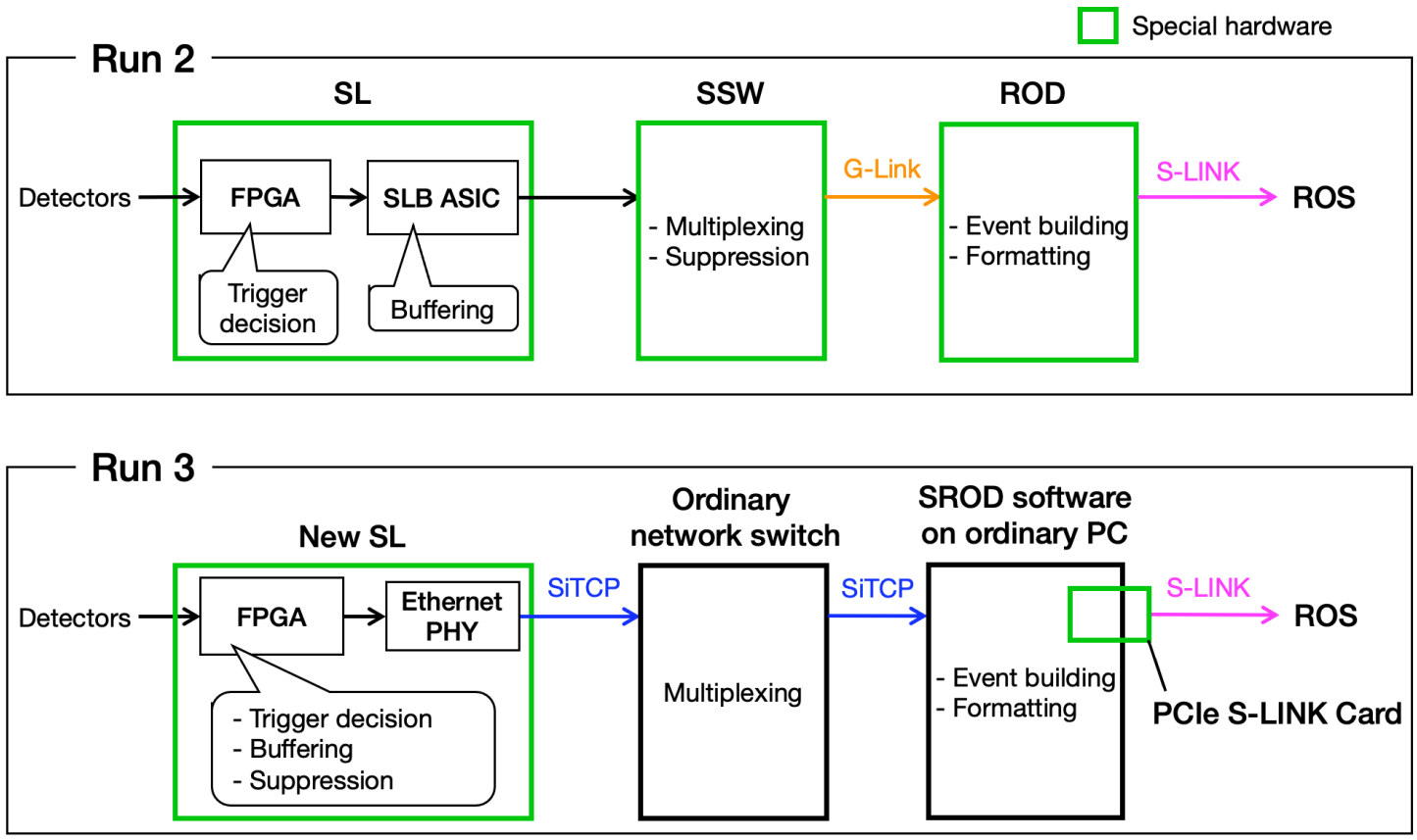


図 4.23　Level-1 ミューオンエンドキャップトリガーのデータ読み出しシステムの Run 3 アップグレード。Run 2 では特別に開発したハードウェアを使用していたため、NSW や RPC BIS78 の導入によるトリガーデータの増加に容易に対応できなかった。Run 3 ではファームウェアやソフトウェアベースの、柔軟性に長けた読み出しシステムを使用する。特に SROD PC 上で走る Software-based Readout Driver (SROD) という読み出しソフトウェアを用いてイベントビルディングを行い、PCIe S-LINK Card を用いて Level-1 トリガーレート 100 kHz でトリガーデータを ROS に送信する。

表 4.1　SROD PC の主な仕様。

	概要
マザーボード	Super X11SPW-TF
CPU	Intel Xeon Gold 6226 プロセッサ (12 コア、2.70 GHz)
メモリ	DDR4-2933 16 GB × 6
ポート	10 GbE ポート ×2、1 GbE ポート ×2
PCIe スロット	PCIe 3.0 x16 スロット ×2、x8 スロット ×2、x4 スロット ×2
OS	CentOS 7.7.1908 (Kernel version: 3.10.0-1062.4.3.el7.x86_64)

ネットワークスイッチ

12 台の New SL と 1 台の TTCFOB の読み出しデータは計 13 本の 1 GbE ケーブルを用いて出力されるが、1 台の PC に 13 本のイーサネットケーブルを接続することは難しい。そのため Run 3 のセットアップではネットワークスイッチを用いることで、これらのケーブルから来るデータをまとめ、1 本のイーサネットケーブルで SROD PC に送信する。今回 USA15 に導入したネットワークスイッチは Ruckus 社の ICX 7450-24 という市販のネットワークスイッチである [34]。このネットワークスイッチは 24 個の RJ45 の受信ポートを持っている。1 本の 10 GbE ケーブルを用いて 12 台の New SL と 1 台の TTCFOB の読み出しデータを SROD PC に送ることができる。

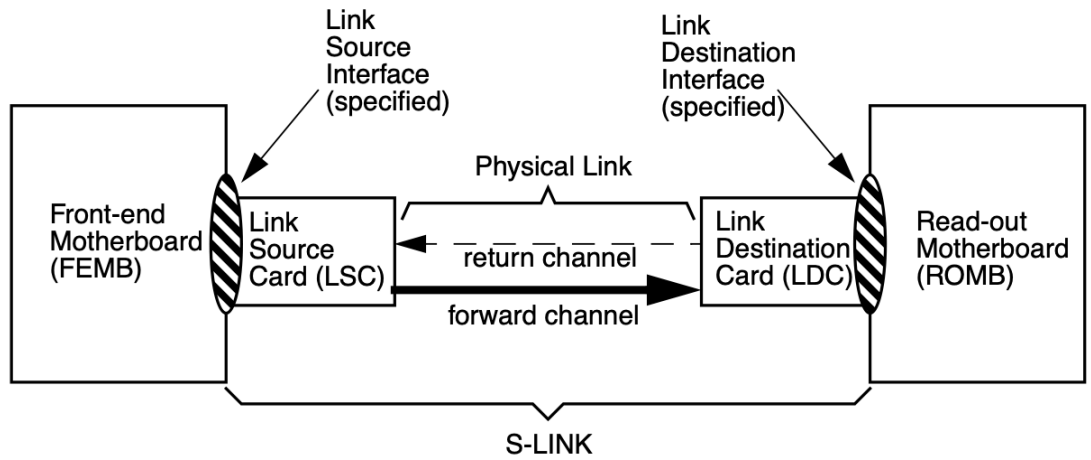


図 4.24　S-LINK 通信の概念図 [35]。

S-LINK 通信と PCIe S-LINK Card

Run 3 のトリガーデータ読み出しパスでは S-LINK[35] という通信規格を用いて、SROD PC から ROS PC に挿入されている RobinNP Card という PCIe カードにデータを送信する。S-LINK は CERN が独自に開発した規格で、データ転送に加えてエラーの検知やデータフローの制御を行う機能を備えている。図 4.24 に S-LINK 通信の概念図を示す。ATLAS 実験の場合、図中の FEMB および LSC が ROD およびその S-LINK 通信を担う部分に、LDC が RobinNP Card に、そして ROMB が ROS PC のマザーボードに対応する。また、forward channel にて ROD の読み出しデータが RobinNP Card に送信され、return channel にて RobinNP Card の状態が ROD 側に送信される。特に RobinNP Card のバッファがいっぱいになってしまった場合、ROD 側からのデータ転送を止めるために XOFF という信号が出力されるようになっている。

図 4.24 を Run 3 の Level-1 ミューオンエンドキャップトリガーのデータ読み出しシステムに見立てた場合、SROD PC のマザーボードが FEMB に対応する。そこで SROD PC 上で走るソフトウェアが S-LINK 規格で RobinNP Card にデータ転送を行えるようにするために、特殊電子回路株式会社と共同で PCIe S-LINK Card という新たな LSC が開発された [36]。図 4.25 にその写真を、表 4.2 にその主な仕様を示す。S-LINK Card にはまず Xilinx Kintex-7 FPGA が搭載されており、これを用いて SROD PC から受信したデータのバッファ処理や RobinNP から受信した XOFF 信号をモニターする。SFP+ ポートには SFP+ 光トランシーバを挿入することで、RobinNP と S-LINK 規格を用いた光通信ができる。また、4 つある LEMO コネクタのうち図 4.25 の最も上にあるものを用いて、オープンドレイン方式の busy 信号を出力できるようになっている。USA15 に新たに設置された 6 台の SROD PC のそれぞれにはこの PCIe S-LINK Card が挿入されており、RobinNP Card との間の光ケーブルも接続済みである。

表 4.2　PCIe S-LINK Card の主な仕様。

	概要
型番	COSMOK0
FPGA	Xilinx Kintex-7 FPGA XC7K160T-2FFG676C
オンボードメモリ	DDR3 SDRAM 1 GB
PCIe	PCIe 2.0 x4
光ポート	SFP+ ポート ×3
LEMO I/O	ECL 入力 ×2、ECL 出力 ×1、オープンドレイン出力 ×1

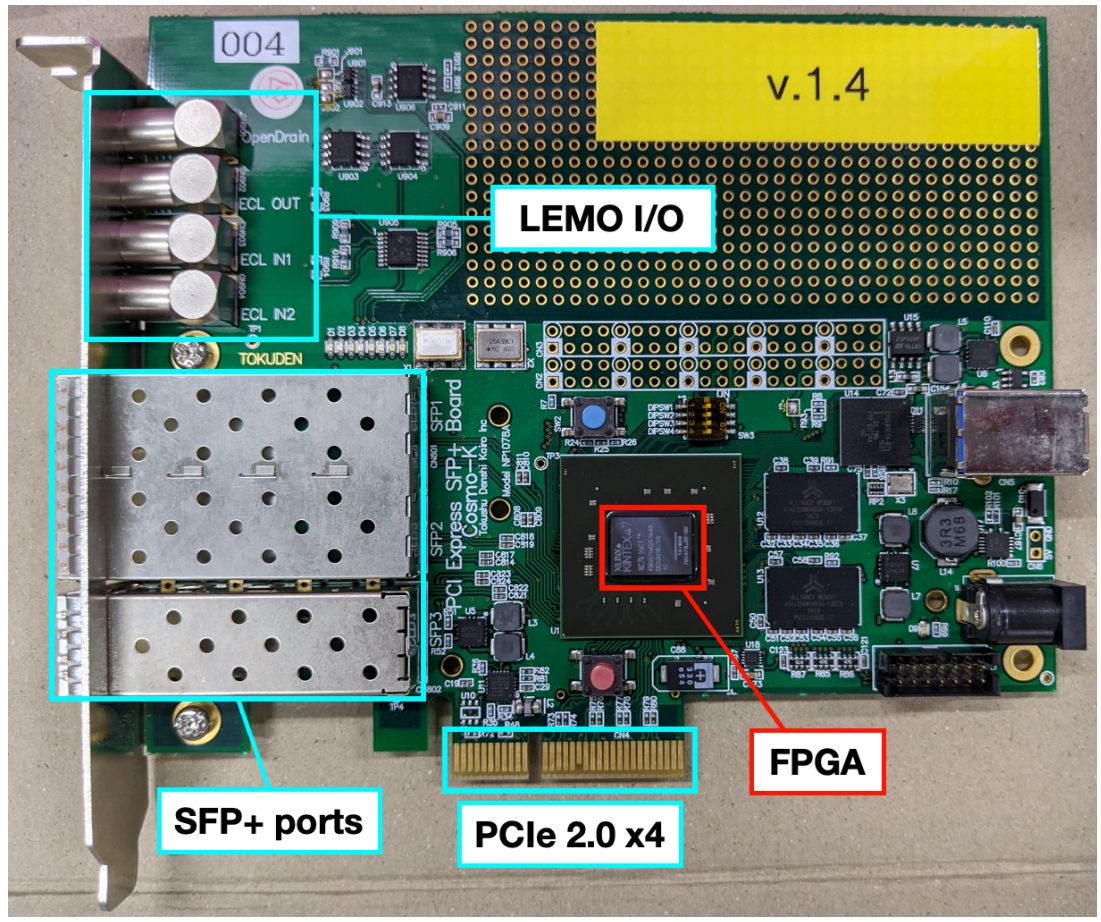


図 4.25　Run 3 に向けて新たに開発された PCIe S-LINK Card。

第 5 章

Level-1 ミューオンエンドキャップトリガーの統合制御の実現

Run 3 から新たに導入する Level-1 ミューオンエンドキャップトリガーシステムでは、6 台の Single Board Computer (SBC) を用いて制御される全 72 台の New SL ボードにおいて、1440 本のリンクを用いて複数種類の検出器のヒット情報を受信し、最終的なトリガー判定を行う。Run 3 においても高い質の物理データを滞りなく収集するためには、Run 2 まで用いていた TGC システムと同期した形で、この新システムの統一的な制御を実現しなければならない。そのためには、操作順序を明確に定義したコンフィギュレーション、想定外の状況から確実に復旧できるようなリカバリー、およびシステム全体の状況把握を可能にする行き渡ったモニタリングなどを系統的に行うことが必要不可欠である。本研究では Level-1 ミューオンエンドキャップトリガーのオンライン制御ソフトウェアを開発することでこれらの機能を実装し、本システムの統合制御を実現した。本章ではこの開発研究について、ならびにこの研究を通して得た知見について議論する。なお SROD に関するソフトウェア開発は本章では議論せず、第 6 章にまとめる。

5.1 ATLAS TDAQ ソフトウェア

ATLAS の各検出器エレクトロニクスは ATLAS 独自の TDAQ ソフトウェアを用いて同期制御される。TDAQ ソフトウェアは主に次に示す 2 種類のソフトウェアから構成される：

- VME バスを用いて検出器エレクトロニクスを制御するためのオンラインソフトウェアアプリケーション
- 検出器エレクトロニクスを制御するために必要なオブジェクトやパラメータを定義する Object Kernel Support (OKS) データベース

前者のソフトウェアアプリケーションは C++ を、後者の OKS データベースは XML を言語として採用している。TDAQ ソフトウェアの動作手順の概要を図 5.1 に示す。各番号の手順の概要は以下の通りである：

1. OKS データベースには ATLAS 検出器の運転に必要なすべてのエレクトロニクスの階層構造や関係性、およびどの PC や SBC で各アプリケーションを走らせるかが定義されている。TDAQ ソフトウェアを立ち上げると、まずは RootController というアプリケーションが指定された PC で起動する。
2. TDAQ ソフトウェアを初期状態に遷移させると、RootController はエレクトロニクスの制御に必要なアプリケーションを走らせるための PC や SBC を呼ぶ。
3. TDAQ ソフトウェアをコンフィギュレーション状態に遷移させると、各 PC や SBC でオンラインソフトウェアアプリケーションが走り始め、エレクトロニクスとの通信が行われる。
4. ソフトウェアアプリケーションの操作に応じて、PC や SBC はエレクトロニクスのコンフィギュレーションに必要なパラメータを OKS から取得する。

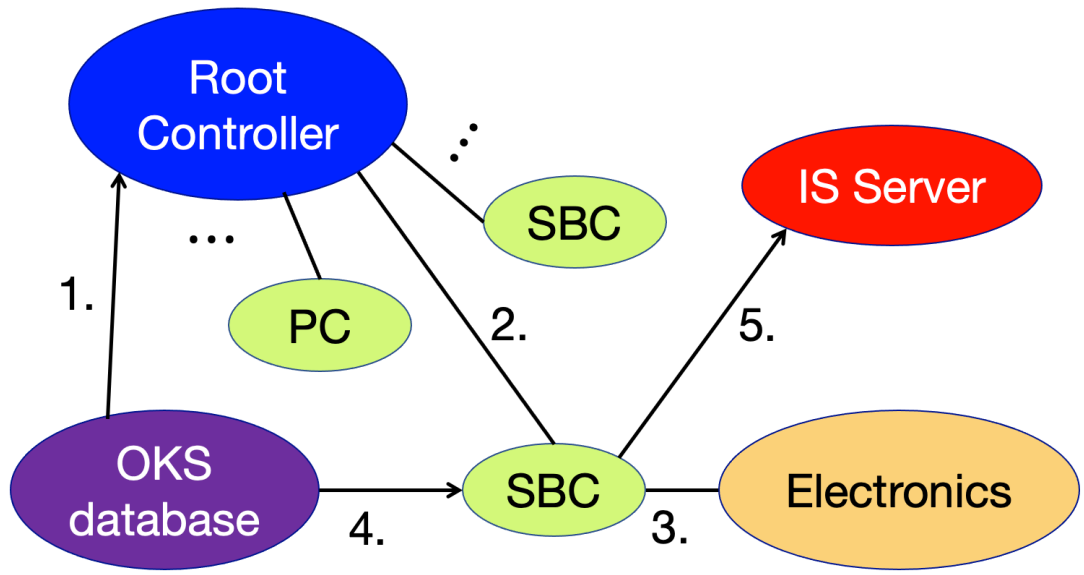


図 5.1　TDAQ ソフトウェアの動作手順。各番号における具体的な説明については本文に記す。

5. Run 中に取得した検出器システムの情報は Information Service (IS) Server に送ることで、外的なデータ可視化ツールを使ってモニタリングすることができる。

このようにしてオンラインソフトウェアアプリケーションと OKS データベースは相互作用して動作する。すなわち ATLAS の検出器エレクトロニクスを正しく制御するためには、これらが互いに整合する形で開発を行う必要がある。

さて、TDAQ ソフトウェアの観点から ATLAS の検出器システムの安定運転を実現するためには、次に挙げる 3 つの事項が柱となる：

- 検出器システムを走らせるための最小単位である “Partition” およびそのシステムの階層構造を定義する “Segments & Resources” を用いたソフトウェアとエレクトロニクスの系統的な制御
- システム内、システム間の同期制御を意識した、運転状態の正しい遷移 (state transition)
- システムの異常を検知・防止するための堅牢なオンラインモニタリング

本節ではこれらの 3 つの事項について小節を設けて具体的に説明する。最後に、検出器システムの運転を行う際に使用する Run Control Panel の概要をまとめる。

5.1.1　OKS データベースの概要と検出器システムの階層構造

ATLAS の検出器システムは複数の検出器および様々なエレクトロニクスから構成される大規模システムである。OKS データベースは、それらのエレクトロニクスの階層構造を正確な定義および運転パラメータの保持と管理を担う。コミッショニングを進める上で必要な試運転を行う場合、あるいは実際の物理ランの最中にとあるハードウェアが壊れた場合など、ATLAS の検出器システムを部分的にオン/オフ (enable/disable) したい状況が多々ある。そのような状況では、検出器エレクトロニクスの enable/disable の状態に応じてソフトウェアアプリケーションの動作も切り替える必要がある。さらに検出器システムの階層に応じて、とあるエレクトロニクスの enable/disable の状態を異なるエレクトロニクスに伝播させたい場合もある。ATLAS TDAQ ソフトウェアでは、OKS データベースに制御対象であるエレクトロニクスやアプリケーションに対応するオブジェクトとそれらの階層構造を定義することで、上記のように同期した形で系統的な制御を行うことが要求される。本研究で実現した Level-1 ミューオンエンドキャップトリガーの統合制御もこれに準ずる。すなわち本小節では、OKS の概要と OKS を用いて検出器システ

```
<class name="TGCFE_CciModule">
 <superclass name="Resource"/>
 <attribute name="Side" type="enum" range="A,C" init-value="A"/>
 <attribute name="Sector" type="enum" range="01,02,03,04,05,06,07,08,09,10,11,12,13,14,15" init-value="01
 <attribute name="Environment" type="enum" range="Global,Local,MuonCombined,SLBTP,ASDTP,Testbench,Debug"
 <relationship name="DBSetting" description="Database Setting for Oracle" class-type="TGCFE_DBSetting" lo
 <relationship name="TTCrxChips" class-type="TGCFE_TTCrx" low-cc="one" high-cc="many" is-composite="no" i
</class>
```

in muons/schema/tgcfe.schema.xml

```
<obj class="TGCFE_CciModule" id="Cci-A01_Physics">
 <attr name="Side" type="enum">"A"</attr>
 <attr name="Sector" type="enum">"01"</attr>
 <attr name="Environment" type="enum">"Global"</attr>
 <rel name="DBSetting">"TGCFE_DBSetting" "TGCFE_DBSetting_A_Physics"</rel>
 <rel name="TTCrxChips" num="4">
  "TGCFE_TTCrx" "TGCFE-A01_TTCrx_M1_phi01"
  "TGCFE_TTCrx" "TGCFE-A01_TTCrx_M1_phi23"
  "TGCFE_TTCrx" "TGCFE-A01_TTCrx_M3_phi01"
  "TGCFE_TTCrx" "TGCFE-A01_TTCrx_M3_phi23"
 </rel>
</obj>
```

in muons/segments/TGC/FE/TGCFE-A_Segment.data.xml

図 5.2　OKS の schema ファイルと data ファイルの一例。上では schema ファイル内で TGCFE_CciModule class を定義しており、下では data ファイル内で TGCFE_CciModule class の Cci-A01_Physics という名前のオブジェクトを定義している。名前の通り、これらは USA15 に設置されている CCI モジュールを制御するために用意されている。

ムの階層構造を構築するために必要な事項についてまとめる[*1]。

OKS データベース

Object Kernel Support (OKS) は前述したように、ATLAS TDAQ ソフトウェアのデータベースである。OKS には検出器エレクトロニクスを制御するために必要なコンフィギュレーションパラメータや、後述する検出器システムの階層構造が定義してある。OKS を用いる最大の利点は、データベースの読み込みに数秒程度しか要さないことである。頻繁に変更するコンフィギュレーションパラメータに関しては、OKS にその値を用意し、ソフトウェアアプリケーションではそれを参照するように実装することが好ましい。このようにすることで、パラメータの変更のためだけにソフトウェアアプリケーションを再コンパイルするような状況を避けることができ、より効率的に試験を進めることができる。すなわち OKS データベースを用いることで洗練されたオンライン制御システムを実現できる。

OKS 上のファイルは schema ファイルと data ファイルの 2 種類から構成される。図 5.2 にそれらの例を示す。Schema ファイルではオブジェクトを定義するために用いる class を定義する。Data ファイルでは、schema ファイルで定義した class を用いて具体的なオブジェクトを定義する。オブジェクトには固有の id をつける必要があり、この id によってオブジェクトの識別が行われる。各オブジェクトが持つコンフィギュレーションパラメータは data ファイルに定義することで、ソフトウェアアプリケーションから参照できる。

OKS を用いた検出器システムの階層構造の定義

検出器システムの階層構造を正しく定義するためには、図 5.3 に示す “Partition”、“Segment”、“Resource”、や “ResourceSet” などの class を用いる。Partition は検出器システムを統合運転するための最小単位であり、その統合運転に必要なシステムをすべて含む。すなわち検出器システムを運転するということは、Partition を走らせることと同義である。実際に物理データを収集するために使用する Partition (ATLAS Partition) は 3.2 節で説明した検出器のエレクトロニクスをすべて含まねばならない。しかし ATLAS 検出器の一部のみを用いて試運転を行いたい場合などは、各サブシステムが定義した Partition を用いる。こうすることでシステム間の干渉を防ぎ、サブシステムごとに試験や開発を並行して進めることができるようになっている。

Partition の配下にある Segment は独立して制御が可能なシステムを定義する。TGC を例として挙げると、TGC の検出器システムは他の検出器システムとは独立して制御ができるので、Segment として定義される。また TGC の A-side と C-side も別々に制御が可能なので、それぞれ別の Segment として定義される。各 Segment は PC や

[*1] OKS データベースにおけるオブジェクトの詳細は付録 C.1 に記すので、適宜参照していただきたい。

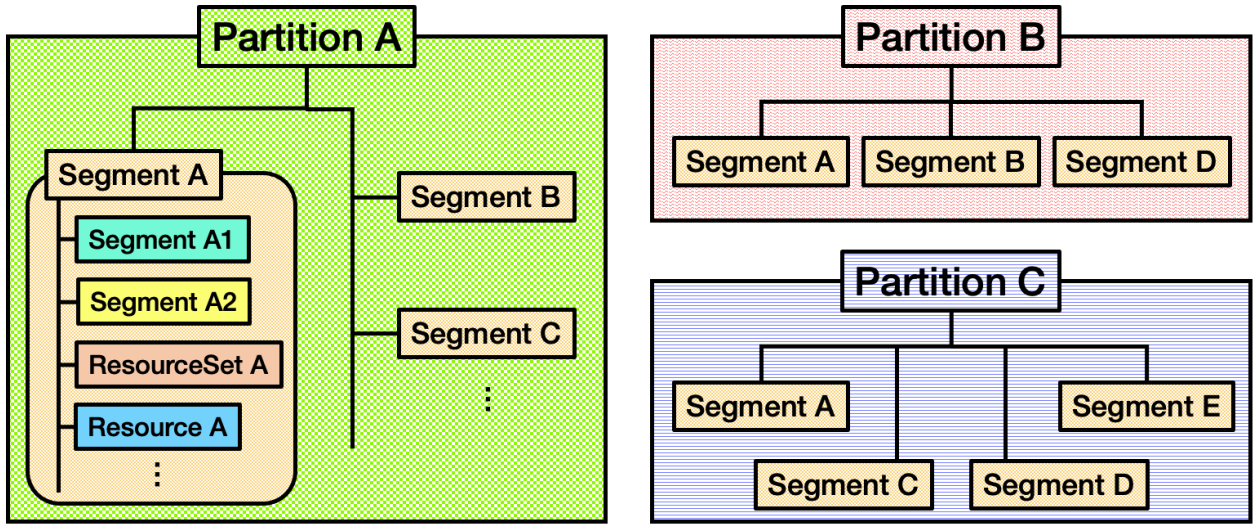


図 5.3　OKS を用いた検出器システムの階層構造の模式図。各 Partition には 1 つあるいは複数の Segment が含まれており、各 Segment には Segment、ResourceSet、Resource などが含まれる。各検出器を運転するためには適切な階層構造を定義しなければならない。

SBC 上で走る Run Control Application というアプリケーションによって Partition と同期して制御される。

Resource は検出器システムの階層構造における最小単位で、1 台のエレクトロニクスや 1 つのアプリケーションに対応する。Resource を enable/disable することで、対応するエレクトロニクスやアプリケーションが Partition を走らせるときに使用されるか否かを指定できる。また ResourceSet を用いて Resource をまとめることもできる。ResourceSet を enable/disable するとそれに含まれる Resource も同様に enable/disable される。さらに、ResourceSet には ResourceSetAND や ResourceSetOR という属性を持たせることができる。ResourceSetAND では、配下にある Resource がすべてdisable された場合、自身も自動で disable される。ResourceSetOR では、配下にある Resource が1 つでもdisable された場合、自身も自動で disable される。

すなわち検出器システムを正しく制御するためには、、図 5.3 に示したように、Segment、Resource、ResourceSet などを駆使してシステムの階層構造を定義することが必要条件となる。また ResourceSetAND や ResourceSetOR などを使い分けることで、Resource の enable/disable の伝播が正しく行われるようにする必要がある。物理データを取得する運転ではすべての Resource を enable するはずだが、コミッショニングにおける利便性やヒューマンエラーなどの不測の事態に備えて、そのような実装を心がけなければならない。

Segment や Resource などを用いて定義される階層構造のことを “Segments & Resources” と呼ぶ。基本的には、Segments & Resources の開発は各サブシステムが独自に開発している Partition (Standalone Partition) を用いて行うことができる。それぞれで開発した階層構造を物理データを取得するための ATLAS Partition に実装する際は、その階層構造の最上位にある Segment を ATLAS Partition に含めれば良い。本研究においても、TGC システムの Standalone Partition である “DAQSlice Partition” を用いて Level-1 ミューオンエンドキャップトリガーと TGC の Segments & Resources を開発し、のちにそれを ATLAS Partition に組み込むことで開発した統合制御システムのバリデーションを行なった。

5.1.2　TDAQ の Run Control State と State Transition

ATLAS 検出器は数多くの検出器システムを内包しており、これらは独立した CPU を用いて制御される。しかし安定して正しい物理データを収集するためには、これらの検出器システムを同期して制御し、システム間の通信リンクを適切な手順で確立しなければならない。このような形で各検出器システムの統合制御を実現するために、ATLAS TDAQ ソフトウェアは Run Control Finite State Machine (FSM) という有限状態機械を採用している。

Run Control FSM は運転の状態を表す Run Control (RC) State と、その状態間の遷移を引き起こす Run

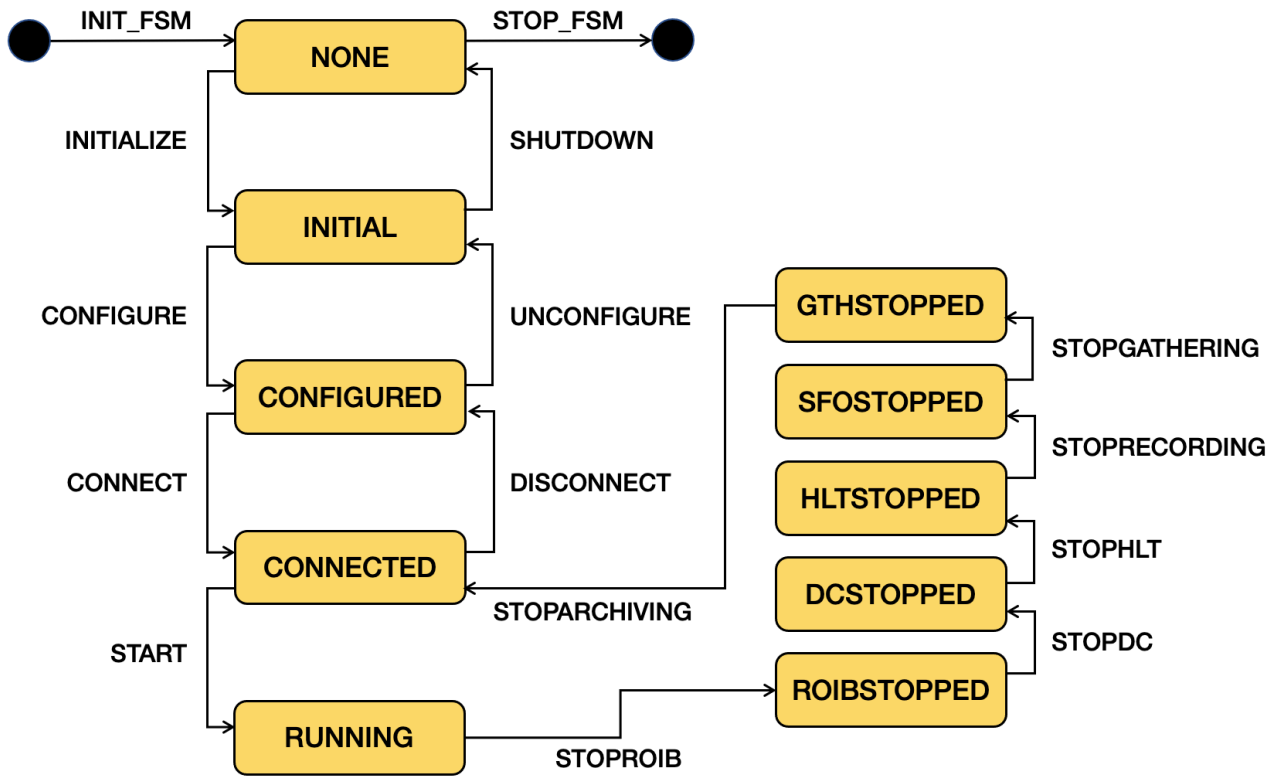


図 5.4　TDAQ ソフトウェアの Run Control FSM。それぞれの箱が Run Control State を、矢印が Run Control Command を表している。

Control (RC) Command から構成される。図 5.4 に TDAQ ソフトウェアの Run Control FSM の全体像を示す。Partition を起動すると、INIT_FSM という RC Command によって Run Control FSM が起動する。FSM 起動直後は検出器システム自体は一切動作していない状態にあるので、NONE という RC State にある。そこから INITIALIZE、CONFIGURE、CONNECT という RC Command を順番に送って RUNNING という RC State にすることで、検出器システムが運転状態になり、データ収集が行われる。システムの運転およびデータ収集を正常に止める際にはいくつかの RC State を経て、再び NONE という RC State に持っていく必要がある。各 RC Command の説明は表 5.1 に記す。

RC Command は RootController によって各 Segment を制御している Run Control Application に配られる。Run Control Application がある RC Command を受け取ると、それが制御する Segment の配下にある Segment や Resource にもその RC Command が配られる。各 Segment や Resource を制御するソフトウェアアプリケーションは受け取った RC Command に応じて、該当する関数を呼ぶようになっている。すなわち検出器システム全体を正しく統合制御するためには、各モジュールのソフトウェアアプリケーションが各 RC Command を受け取った際にどのような操作を行うかという、Run Control の状態遷移 (state transition) を正しく定義しなければならない。特にサブシステム内の各モジュール間およびサブシステム間の同期を実現するには、必要な操作を順序立てて行う必要がある。本研究では Run 3 に向けて Level-1 ミューオンエンドキャップトリガーと TGC の新しい state transition を考案し、実装した。詳しくは 5.4 節で説明する。

5.1.3　オンラインモニタリング

検出器システムの正常動作は安定して物理データを収集するための必須条件である。システムの誤動作あるいはその兆候を検知するために、各ボードに使われているモジュールやチップの状態のモニタリングは欠かせない。このため TDAQ ソフトウェアには様々なオンラインモニタリングサービスが用意されている。そのうち本研究で使用したものを以下で説明する。

表 5.1　RC Command の概要。検出器システムの運転を行う際は、自動的に複数の RC Command を連続して行う CONFIG、UNCONFIG、STOP も使われる。

RC Command	遷移元の RC State	遷移先の RC State	RC Command の説明
INIT_FSM		NONE	RC FSM を起動する。
STOP_FSM	NONE		RC FSM を終了する。
INITIALIZE	NONE	INITIAL	RootController を起動させ、システムを制御するために必要な PC や SBC を呼ぶ。
SHUTDOWN	任意の RC State	NONE	システムの運転を終了する。正常な終了は INITIAL から行うが、任意の状態から強制終了を行うことも可能である。
CONFIGURE	INITIAL	CONFIGURED	OKS からコンフィギュレーションパラメータを取得して、検出器システムを運転するために必要な準備を行う。
UNCONFIGURE	CONFIGURED	INITIAL	–
CONNECT	CONFIGURED	CONNECTED	サブシステム内の各モジュール間やサブシステム間の接続を確立する。
DISCONNECT	CONNECTED	CONFIGURED	–
CONFIG	INITIAL	CONNECTED	自動で CONFIGURE と CONNECT を連続して行う。
UNCONFIG	CONNECTED	INITIAL	自動で DISCONNECT と UNCONFIGURE を連続して行う。
START	CONNECTED	RUNNING	システムの運転およびデータ収集を開始する。
STOPROIB	RUNNING	ROIBSTOPPED	RoIB の動作を停止する。
STOPDC	ROIBSTOPPED	DCSTOPPED	DCM の動作を停止する。
STOPHLT	DCSTOPPED	HLTSTOPPED	HLT の動作を停止する。
STOPRECORDING	HLTSTOPPED	SFOSTOPPED	データ記録を停止する。
STOPGATHERING	SFOSTOPPED	GTHSTOPPED	データ収集を停止する。
STOPARCHIVING	GTHSTOPPED	CONNECTED	検出器システムの情報の発行を停止する。
STOP	RUNNING	CONNECTED	自動で STOPROIB から STOPARCHIVING までの停止用 RC Command を連続して行う。

Information Service (IS)

Information Service (IS) は、各サブシステムが定義したパラメータをオンラインでモニターするためのモニタリングサービスである。各サブシステムは TDAQ グループが用意している IS 用の API を用いることで、自身のシステムで取得したパラメータをソフトウェアアプリケーション経由で IS に定期的に送信することができる。歴史的な理由により、IS は $O(1\text{–}10)$ 秒という比較的低いレートで各パラメータの最新のスナップショットを送信する方式を採用しており、データが更新されると過去の情報は失われてしまう。また、IS に送信したデータはソフトウェアアプリケーションを用いて逆に取得することも可能である。

送信したパラメータの値は図 5.5 に示す IS の GUI を用いてオンラインでモニターすることができる。使用している Partition を選択し、モニターしたいパラメータが属するサーバー名を選ぶことで、そのパラメータの値の最新

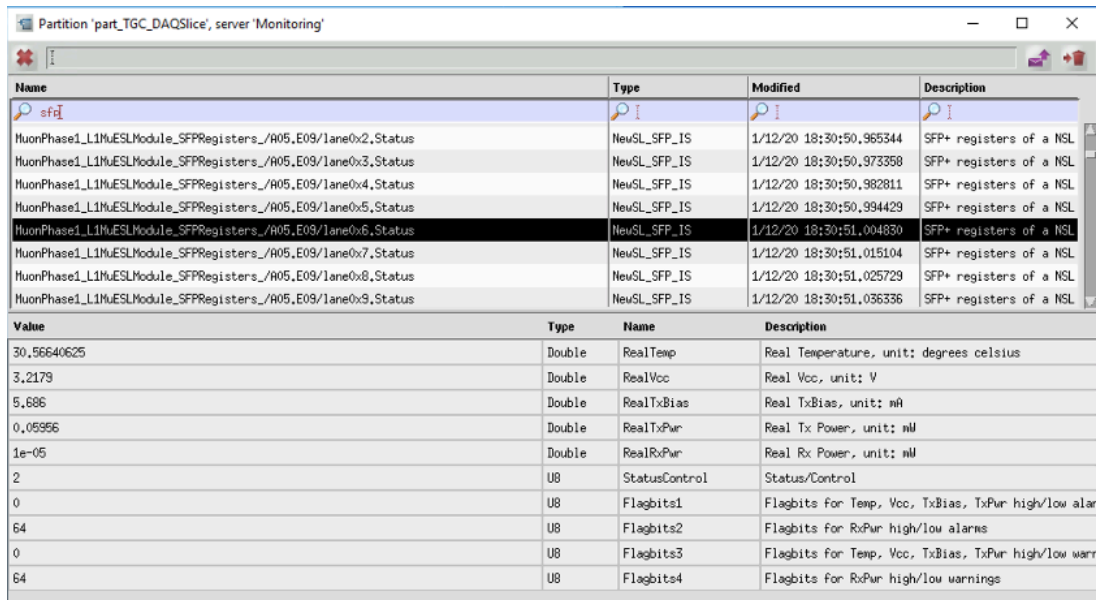


図 5.5　IS の GUI。オンラインソフトウェアで IS サーバーに送ったパラメータはこの GUI を用いて監視できる。

time	sev	msgID	application	text	host
19:36:30 Dec 04 2020	ERROR	TGCSROD::Issue	SROD-EventBuilder-A2	ID mismatch between SROD-TTCFanoutModule-A2 and SROD-SLModule-A08-F08. TTC-L1ID [0x0], SL-L1ID [0x0] TTC-BCID [0x228], SL-BCID [0x225] : BCID_MISMATCH	pc-tgc-rod-lvl1-02.cern.ch

図 5.6　ERS メッセージの一例。Time はそのメッセージの発行時間、severity は重要度、msgID はメッセージの種類、application は発行元のアプリケーションの名前、text は内容、そして host は発行元のアプリケーションが走っていた PC や SBC の名前を表している。

のスナップショットが見られる。しかし瞬間的な情報しか得られないため、IS のみを用いたオンラインモニタリングは時には利便性に欠ける。

このため、IS に送ったデータを永続的に保持するために Persistent Back-End for ATLAS information System of TDAQ (PBEAST) という、モニタリングサービスが用意されている。PBEAST に保持されたデータは Grafana というウェブモニタリングツールを用いることで、ヒストグラムなどを用いたユーザーフレンドリーな形で、オンラインでモニターすることができる。本研究の一環として、Level-1 ミューオンエンドキャップトリガーのソフトウェアアプリケーションに一部モジュールのモニタリング機能を実装し、IS や Grafana を用いたモニタリングを可能にした。詳細は 5.5 節に記す。

Error Reporting Service (ERS)

Error Reporting Service (ERS) とは、検出器システムの異常を中心とした運転状態をメッセージ形式で表示するためのモニタリングサービスである。システムに異常が検知された場合に備えて、各サブシステムは ERS を用いてエラーメッセージを表示するようにソフトウェアアプリケーションに実装しなければならない。図 5.6 に示すように、ERS のメッセージは time、severity、msgID、application、text、host の 6 つの項目から構成される。特に severity はそのメッセージの重要度を表す項目であり、各サブシステムでは状況に応じて適切な severity のメッセージを発行するように心がけなければならない。表 5.2 に severity の種類を示す。

IS や ERS 以外にも衝突事象をオンラインで表示する Event Monitoring Service (EMS) や、自動で Root を使ったヒストグラムを生成する Online Histogramming Service (OHS) などのオンラインモニタリングサービスが用意されている。

表 5.2　ERS の severity の一覧。

Severity	概要
Fatal	メッセージの発行元が使用できる状態にないことを示す。Fatal が出た場合はその発行元をリセットあるいは物理的に修理することで復帰させない限り、次の RC State には進めない。ソフトウェアアプリケーションと OKS の間に不整合がある場合や、ハードウェアが故障した場合などに表示される。
Error	メッセージの発行元が要求された処理をできなかったことを示す。Fatal とは異なって次の RC State に進むことは可能だが、正常な運転中に出るメッセージではない。
Warning	メッセージの発行元が要求された処理は実行できたが、注意すべき点を検知したことを示す。壊れたデータを検知して復元したなどのデータの質に関わる事項や、ディスクの空き容量が少なくなったなどの error や fatal を引き起こしうる事項を検知した場合に使用される。
Information	検出器システムの正常運転に影響を及ぼさない、重要度が低い情報を表示するために使用される。

5.1.4　Run Control Panel を用いた検出器システムの運転

検出器システムを運転するために必要な Partition を起動すると、図 5.7 に示す GUI が立ち上がる。これは Run Control Panel と呼ばれ、実際に ATLAS の検出器システムを運転するために使われる。上部の緑枠で囲っている部分は起動した Partition の名前を表している。左側の赤枠内にある RC Command のボタンを押すことで、図 5.4 の Run Control FSM に応じた検出器システムの制御が行われる。また赤枠の上にある Commit & Reload ボタンを押すことで、OKS データベースを再読み込みすることができる。中央の青枠内には Run Control Application の階層構造およびそれぞれの RC State が表示されている。下のピンクの枠内には ERS のメッセージが表示されており、運転中に生じたエラーなどを素早くモニタリングできるようになっている。

図 5.7 の青枠の上にある Segments & Resources タブを押すことで、図 5.8 に示すような Segments & Resources の階層構造を見ることができる。ここから各 Segment や Resource を enable/disable することができる。この enable/disable の操作は OKS のデータベースに変更を加えるため、enable/disable を行なった場合はデータベースの再読み込みが必要になる。

5.2　オンラインソフトウェアによる Level-1 ミューオンエンドキャップトリガーと TGC の制御の全体像

5.1 節でも議論したように、検出器システムの系統的な制御を実現するためにはオンラインソフトウェアの開発は欠かせない。そこで本研究では Run 3 から新たに導入される Level-1 ミューオンエンドキャップ (L1MuE) トリガーを、Run 2 でも用いていた TGC システムと同期する形で統合制御するために、L1MuE トリガーおよび TGC のオンラインソフトウェアを開発した。この開発研究の詳細に入る前に、本節では L1MuE トリガーおよび TGC のオンラインソフトウェアに要求される機能、それらの制御に必要な PC と SBC、およびそれらのソフトウェアの全体像について説明する。

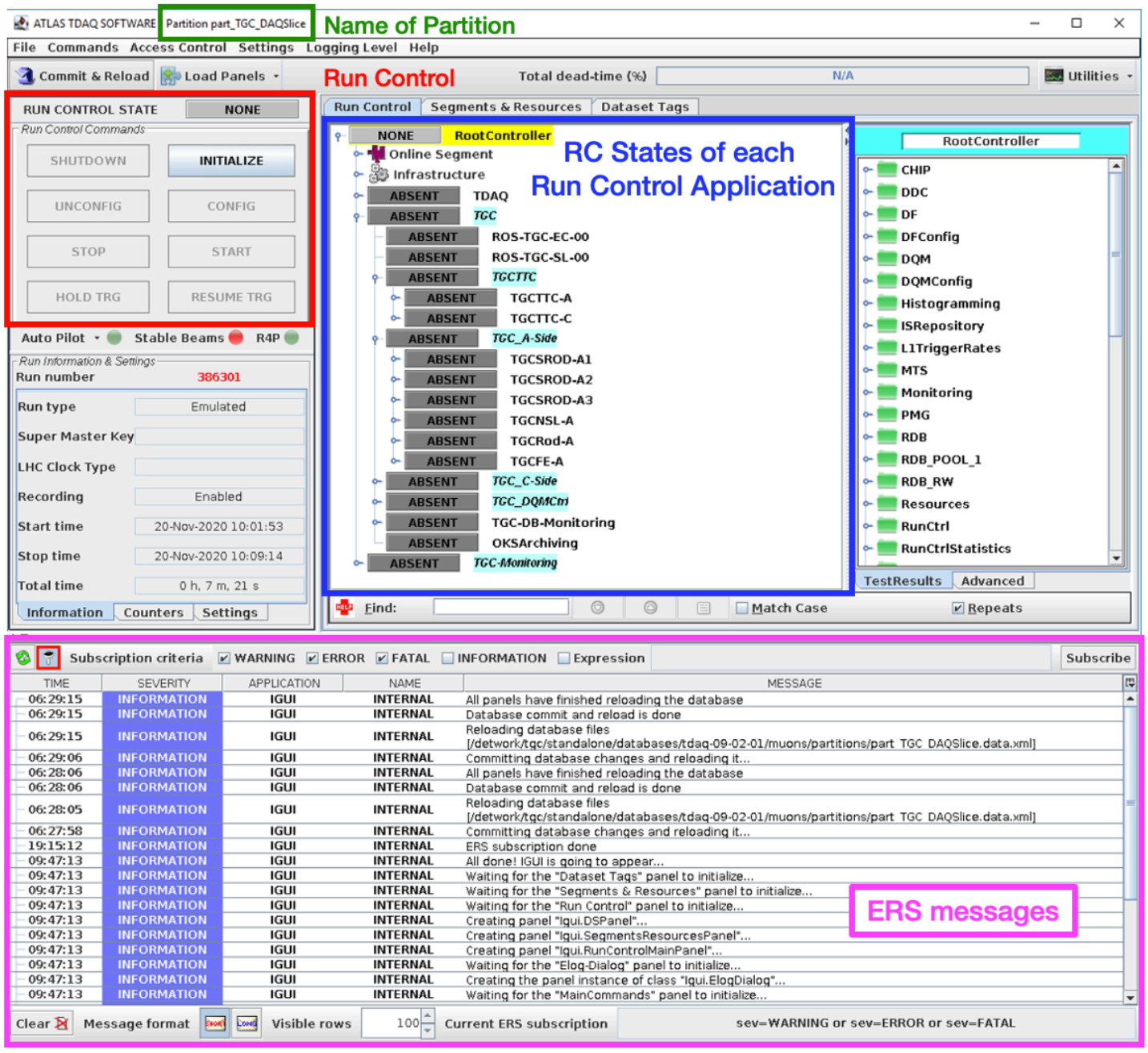


図 5.7　TDAQ ソフトウェアの Run Control Panel。

5.2.1　Level-1 ミューオンエンドキャップトリガーと TGC のオンラインソフトウェアに要求される機能

Run 3 においても物理データを安定して収集するためには、新たに導入される L1MuE トリガーの統合制御を実現しなければならない。その統合制御の実現にあたって、以下の機能が達成される必要がある：

1. TDAQ ソフトウェアを用いた、New SL を中心とした New SL クレートのすべてのエレクトロニクスの同期制御
2. New SL クレートのすべてのエレクトロニクスの TTC 信号との同期
3. 全 72 台の New SL が持つ計 1440 本の光リンクを用いた、周囲の検出器システムとの通信の確立
4. L1MuE トリガーが想定外の状態に陥った場合、素早い reconfiguration によってシステムを復旧する機能
5. New SL クレートのエレクトロニクスの状態を監視するための行き渡ったモニタリング

1. は新しい L1MuE トリガーのエレクトロニクスを ATLAS の検出器システムの一部として走らせるための最低条

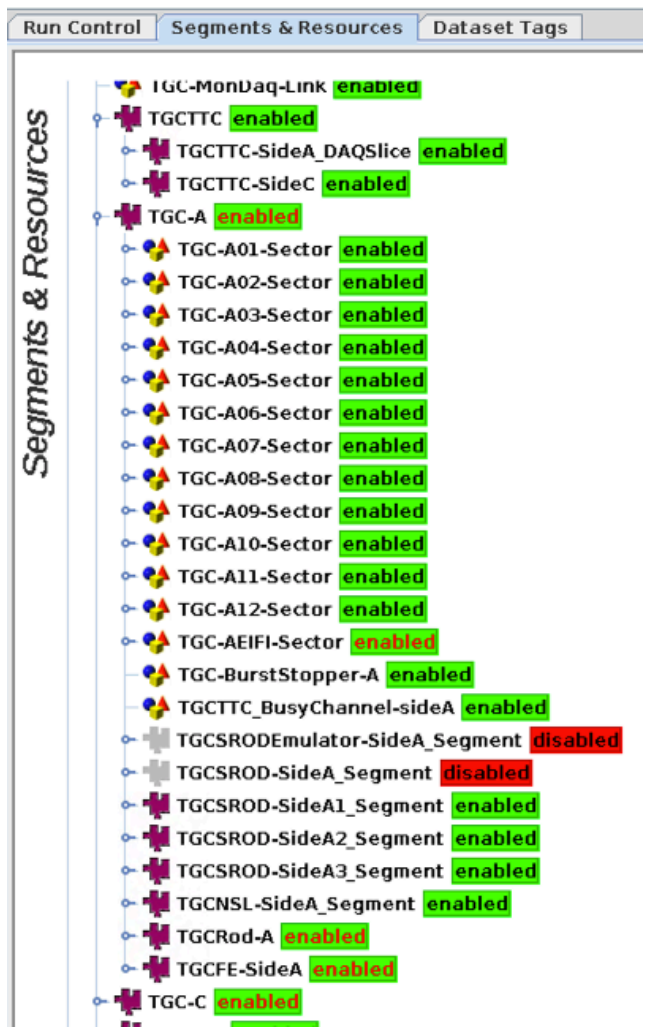


図 5.8　Run Control Panel から見た Segments & Resources。パズルのピースが Segment を、赤青黄の立体図形のマークが Resource や ResourceSet を表している。

件であり、これが達成されない限り USA15 にある New SL を用いた統合制御試験は一切できないので、非常に重要な開発事項である。2. と 3. は新しい L1MuE トリガーと TGC を含む大規模システムを同期して制御し、正しいタイミングでトリガーを出力するために必要な機能である。4. は万が一検出器システムが正しく動作しなくなった場合、物理データの損失を最小限に抑えるために欠かせない機能である。5. はシステムがそもそも正しく動作しないような状況に陥る前にその兆候を検知し、問題の発生を防ぐために必要である。本研究ではこれらの機能を実装・達成するべく、L1MuE トリガーおよび TGC のオンラインソフトウェアの開発に取り組んだ。開発の詳細については 5.3 節から 5.5 節で議論する。

5.2.2　制御に用いる PC と SBC

本小節では L1MuE トリガーと TGC のオンラインソフトウェアを走らせ、各エレクトロニクスを制御するために必要な PC と SBC についてまとめる。図 5.9 に USA15 における PC と SBC およびそれらが制御するモジュールの配置の模式図を示す。

sbc-tgc-csl-lvl1-0X (X=1–6)

これらは New SL が入っている VME クレート (New SL クレート) を制御するための SBC である。1 つの New SL クレートには 12 台の New SL ボード、1 台の TTCFOB、そして G-Link Converter あるいは Burst Stopper が挿入されている。これらの SBC は VME 通信によって同一クレート内の各モジュールを制御する。図 5.10 に 6 つの New SL クレートの写真を示す。

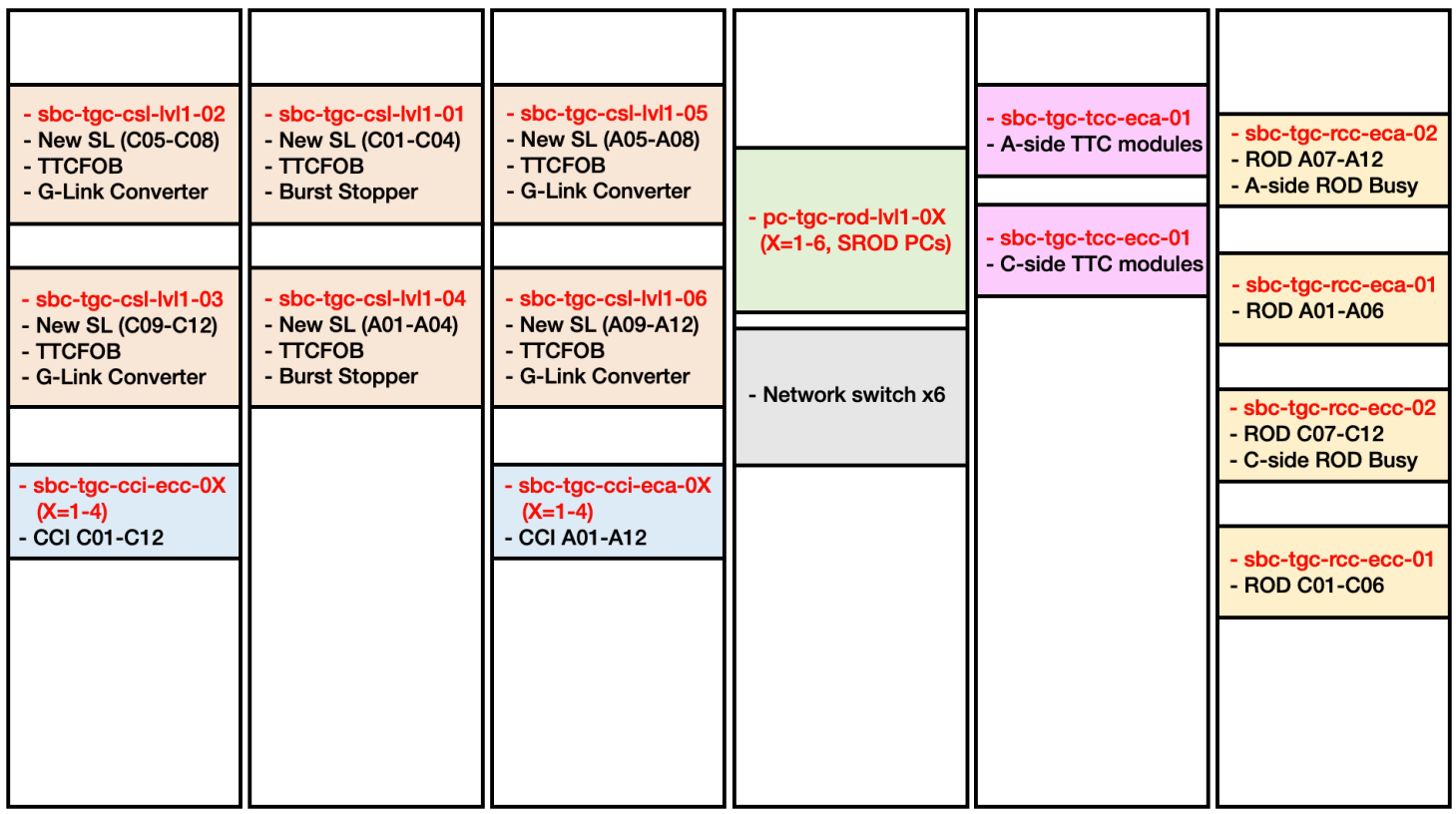


図 5.9　USA15 における L1MuE トリガーの PC と SBC およびモジュールの配置の模式図。52U の 19 インチラックが 6 つ並んでおり、それぞれに 9U や 6U サイズの VME クレートが設置されている。赤字は制御用の PC や SBC を、黒字は代表的なモジュールを示している。各モジュールは 6 つのラックに渡って配置されている。空白の部分には冷却用のファンやケーブルをファンアウトするためのモジュールなどが配置されているが、ここでは省略した。なお 2 台の ROS 用の 19 インチラックは少し離れた場所にあるので、この図には載せていない。

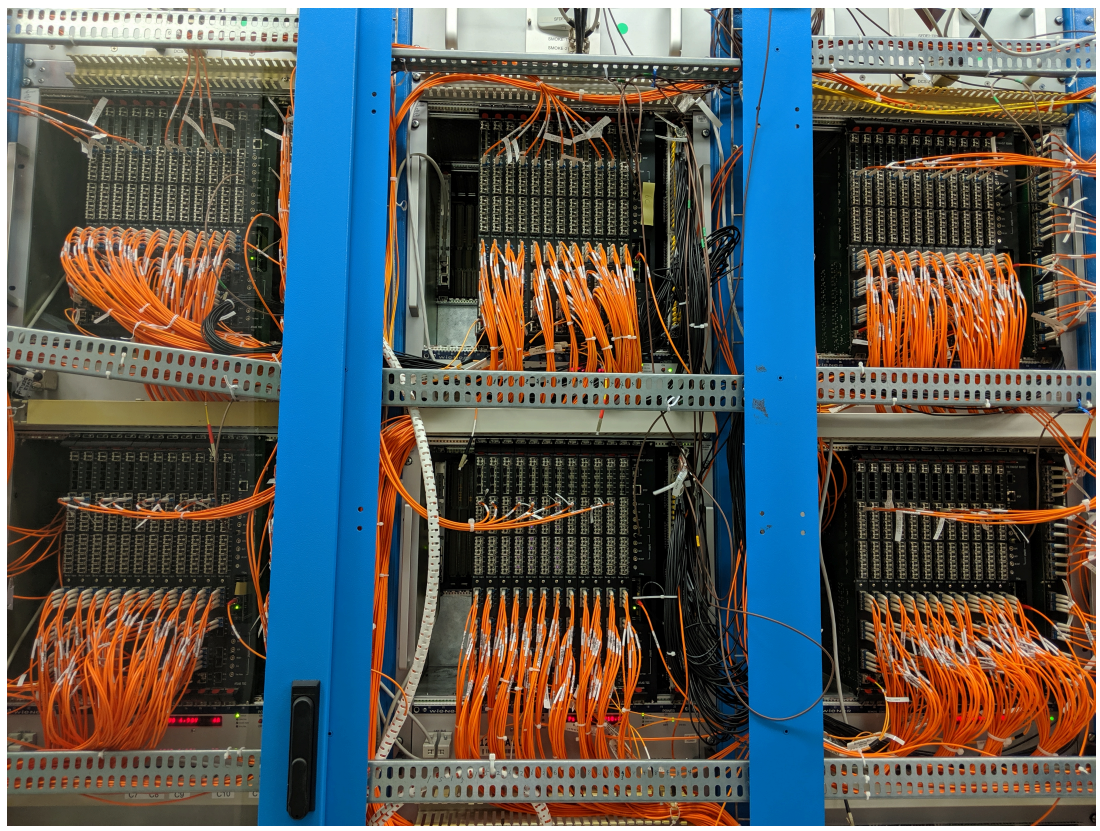


図 5.10　USA15 の 6 つの New SL クレートの写真。1 つの New SL クレートには 1 台の制御用 SBC、12 台の New SL、1 台の TTCFOB、および 1 台の G-Link Converter または Burst Stopper が挿入されている。

sbc-tgc-cci-eca(ecc)-0X (X=1–4)

これらの SBC は同一クレート内の CCI モジュールを通じて、UX15 (ATLAS 実験室) にあるすべてのフロントエンド回路群の制御を担う。4.1.5 節でも述べたように、CCI モジュールは HSC クレート内の HSC モジュールを通じて HPT ボードと SSW を、さらには SSW を通じて TGC 検出器上にある PS Board や Service Patch Panel を制御する。1 台の SBC は 3 台の CCI モジュール、つまり 3 つの 1/12 セクターに対応するフロントエンド回路群の制御を行う。

sbc-tgc-rcc-eca(ecc)-0X (X=1,2)

これらは ROD が入っている VME クレート (ROD クレート) を制御するための SBC である。A-side および C-side のそれぞれについて 2 つの ROD クレートがあり、各 ROD クレートには 6 台の TGC ROD が挿入されている。また、片サイドについて 2 つある ROD クレートの片方には ROD の busy 信号をまとめる ROD Busy Module があるので、その制御も行う。

sbc-tgc-tcc-eca(ecc)-01

これらの SBC は TTC 関係のモジュールが入っている VME クレート (TTC クレート) を制御する。TTC クレートには 4.1.6 節で説明した LTPi、LTP、TTCvi、TTCex、そして Delay Module が挿入されている。

pc-tgc-rod-lvl1-0X (X=1–6)

4.2.4 節ですでに述べたように、これらの PC は Run 3 から導入されるトリガーデータ読み出しソフトウェアである SROD を走らせるために新たに設置されたものである。1 台の SROD PC は 4 つの 1/12 セクター分のトリガーデータをまとめ、それぞれ S-LINK Card を用いて ROS にそれらのトリガーデータを送信する。

pc-tgc-ros-ec-00

これは A-side と C-side を合わせた計 24 台の TGC ROD が送信する読み出しデータを処理するための ROS 用の PC である。この PC には 2 枚の Robin NP Card が挿入されており、1 枚で TGC ROD 12 台分の読み出しデータを受信する。

pc-tgc-ros-sl-00

これは 6 台の SROD PC から送られてくるトリガーデータを処理するための ROS 用の PC で、Run 3 から新たに導入された。1 枚の Robin NP Card を用いて L1MuE トリガーのすべてのトリガーデータを受信する。

5.2.3 制御に必要なオンラインソフトウェアパッケージ

本研究では Run 3 から新たに導入される L1MuE トリガーのオンラインソフトウェアに加え、Run 2 ですでに使用されていた TGC のオンラインソフトウェアの開発も部分的に行なった。この小節ではこれらのオンラインソフトウェアの全体像および実際に開発したパッケージの概要についてまとめる。

図 5.11 に示すように、L1MuE トリガーおよび TGC のオンラインソフトウェアは複数のパッケージから構成される。新たに開発された L1MuE トリガーのオンラインソフトウェアには、New SL クレートを制御するためのパッケージや SROD ソフトウェアを走らせるためのパッケージが用意されている。他方で、TGC のオンラインソフトウェアには、フロントエンド回路の制御、TTC システムの制御、ROD クレートの制御、そして各種モニタリングなどのために用意されたパッケージがある。本研究では赤い点線で囲ったパッケージを中心に開発を行なったので、これらに焦点をおいて説明する。

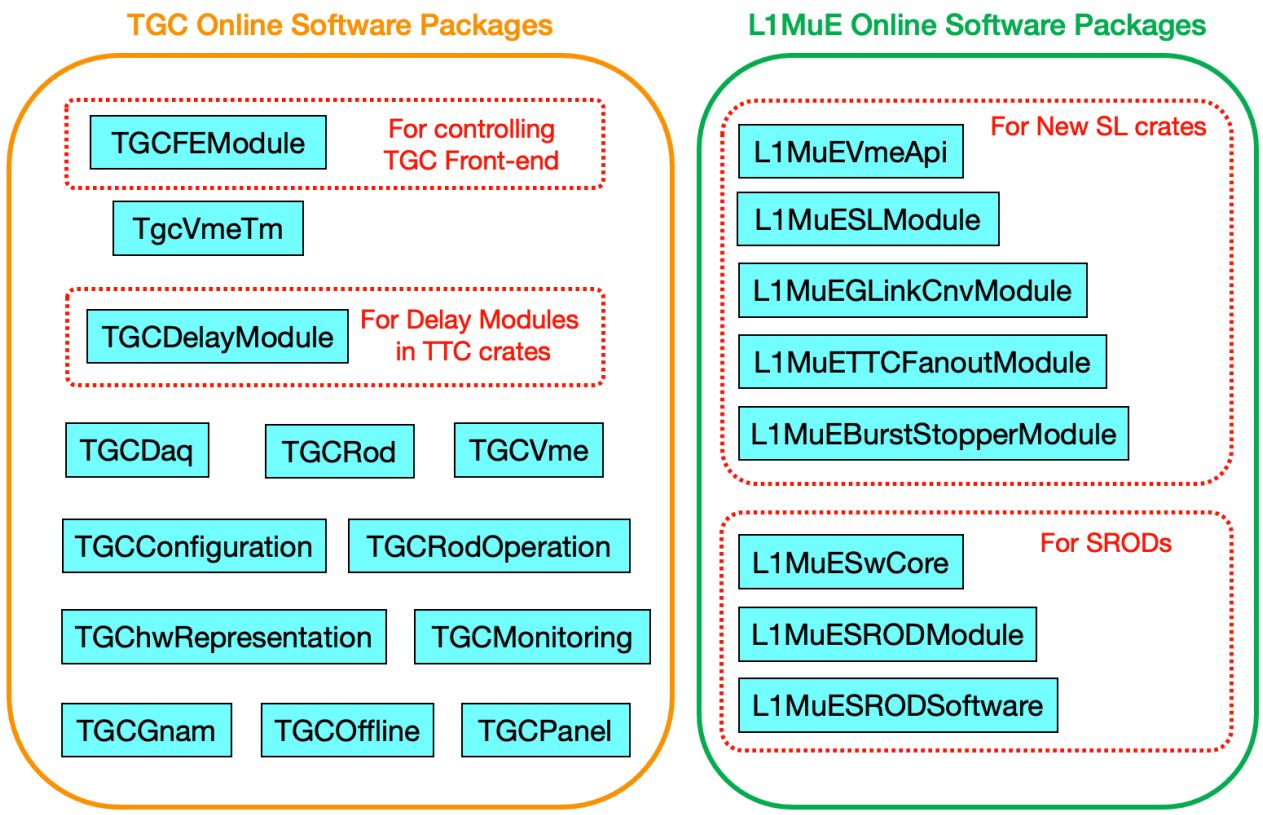


図 5.11　2021 年 1 月現在における L1MuE トリガーおよび TGC のオンラインソフトウェアパッケージの一覧。緑色の枠内にあるのが Run 3 のために新たに追加された L1MuE トリガー用のパッケージで、橙色の枠内にあるのが Run 2 から使われていた TGC 用のパッケージである。現在も開発が進んでいるため、Run 3 開始時には別の構造になっている可能性がある。

New SL クレートの制御に用いるパッケージ群

New SL クレートの制御のために、5 つのオンラインソフトウェアパッケージが用意されている。L1MuEVmeApi は SBC から VME 経由で New SL クレートに挿入されている各ボード上の FPGA などのモジュールにアクセスするための API パッケージである。L1MuESLModule、L1MuEGLinkCnvModule、L1MuEBurstStopperModule、L1MuETTCFanoutModule は SBC 上で走るソフトウェアアプリケーションで、それぞれ L1MuEVmeApi を使って New SL、G-Link Converter、Burst Stopper、および TTCFOB を制御する。これらのソフトウェアアプリケーションの制御は、SBC 上で走る Run Control Application によって逐次的に行われる。これらのパッケージ群を用いて、New SL を中心としたエレクトロニクスにおける運転パラメータのコンフィギュレーション、光リンクの確立、読み出しバッファの初期化、および各モジュールの監視などの機能を実現する。

SROD ソフトウェアを走らせるために用いるパッケージ群

ソフトウェアベースの読み出しシステムである SROD は 3 つのオンラインソフトウェアパッケージから構成される。L1MuESwCore は SROD PC が行うソケット通信や共有メモリの操作を定義する、基本的な API パッケージである。L1MuESRODSoftware は L1MuESwCore を用いて SROD PC 上で走る複数のアプリケーションの具体的な操作を定義する重要なパッケージである。L1MuESRODModule は L1MuESRODSoftware で定義されているアプリケーションの RC State などの管理を担う。これらについては第 6 章でより詳しい説明を与える。

TGCFEModule

TGCFEModule は CCI クレートにある SBC 上で走るソフトウェアで、CCI および HSC モジュール経由で PS Board、SPP、SSW、HPT などの TGC のフロントエンド回路群の制御を行う。CCI クレートにおける VME 通信、HSC クレートにおける VME 通信、そして各モジュールとの通信を行うために、TgcVmeTm という通信用 API を

まとめたパッケージが用意されている。TGCFEModule は TgcVmeTm を使用し、正しい state transition に応じて複数種類のモジュールを系統的に制御しなければならない。このパッケージを用いてフロントエンドの BCID に関する運転パラメータの制御、読み出しバッファの初期化、光リンクの確立などの機能を実現する。

TGCDelayModule

TGCDelayModule は TTC クレートに挿入されている SBC 上で走るソフトウェアで、同じく TTC クレート内にある Delay Module を制御する。検出器システムの運転を開始する前に、TTCrx をリセットするためのブロードキャストコマンドや PS Reset 信号を送る。

5.3 Level-1 ミューオンエンドキャップトリガーおよび TGC の OKS の開発

5.2.1 節でも説明したように、TDAQ ソフトウェアを用いて L1MuE トリガーの新規エレクトロニクスの同期制御を実現するためには、これらのエレクトロニクスに対応するオブジェクトを OKS データベースに定義しなければならない。加えて Run 3 の新しい L1MuE トリガーを Run 2 でも使用していた TGC システムと、5.1.1 節で述べた enable/disable の観点から同期して制御できるようにするために、TGC システム全体の Segments & Resources の構造を吟味し、再編した。このようにして、はじめて 5.2.3 節に示したオンラインソフトウェアパッケージを用いた統合運転ができるようになった。本節ではこれらの OKS の開発についてまとめる[*2]。

5.3.1 Run 3 の新規エレクトロニクスを制御するための OKS の開発

TDAQ ソフトウェアを用いて Run 3 における L1MuE トリガーの統合制御を実現するために、まずは全 72 台の New SL を中心とする New SL クレートのすべてのエレクトロニクスに対応するオブジェクトを OKS データベースに定義する必要がある。ここでは新たに定義した新規エレクトロニクスのオブジェクトの概要についてまとめる。

5.1.1 節で述べたように、OKS のオブジェクトを定義するためには、それが属する class が定義されている必要がある。そこでまず、New SL クレートに挿入されているエレクトロニクスである New SL、TTCFOB、G-Link Converter、Burst Stopper のための class を定義した schema ファイルを用意した。これらの schema ファイルにはそのエレクトロニクス自体を表す class に加え、それらが持つコンフィギュレーションパラメータを定義するための class が定義してある。ここでは新たに定義した 4 つのエレクトロニクスの class のうち、New SL の class のみを表 5.3 で紹介する[*3]。

次に、schema ファイルに定義した class を用いて実際のオブジェクトを定義した data ファイルを用意する必要がある。これらの data ファイルには各エレクトロニクスの制御に用いるコンフィギュレーションパラメータの具体的な値を用意した。図 5.12 に New SL の場合の例を示す。New SL は計 72 台あるので、それに対応した形で 72 個の NewSLBoard という Resource (オブジェクト) を用意した。またそれらの Resource が持つコンフィギュレーションパラメータも New SL ごとに調整できるように、必要な数だけ対応するオブジェクトを定義した。さらに、TTCFOB、G-Link Converter、Burst Stopper についても同様の data ファイルを準備することで、全 6 個の New SL クレートに設置されているすべてのエレクトロニクスを制御するために必要なオブジェクトを定義した。このようにして 27 個のファイルに合計 400 個以上のオブジェクトを用意することで、New SL クレートのソフトウェアアプリケーションが OKS から適切なコンフィギュレーションパラメータを参照して走るための基礎を構築した。

[*2] 詳細な議論はすべて付録 C.3 に記すので、適宜参照していただきたい。
[*3] New SL の OKS class のより正確な定義、および他の 3 つのエレクトロニクスの class の定義は付録 C.3.2 の表 C.1–C.4 に示す。

表 5.3　New SL の OKS class (NewSLBoard) の定義。

コンフィギュレーションパラメータの名称	概要
BoardAddress	New SL ボードの 32 ビット VME アドレス。
BoardType	Endcap または forward。
BoardId	ボードを識別するための ID。E01-E24 あるいは F01-F12 のいずれか。
BitFile	FPGA に書き込むビットファイル。
LUTFile	FPGA に書き込む LUT ファイル。
DelayFPGA	FPGA で受信したヒット信号や TTC 信号にかける遅延時間。
MaskFPGA	FPGA で受信したヒット信号や TTC 信号にかけるマスクパラメータ。
InnerCoincidenceFlagFPGA	FPGA のトリガーパスにおける Inner Coincidence にどの検出器を用いるかを表すフラグ。
ReadoutFPGA	FPGA の読み出しパスのパラメータ。読み出す BC の数や L1 buffer の大きさなどを含む。

```
<obj class="NewSLBoard" id="NewSLBoard-A01-E01">
 <attr name="BoardAddress" type="u32">0x01e00000</attr>
 <attr name="BoardType" type="enum">"Endcap"</attr>
 <attr name="BoardId" type="string">"E01"</attr>
 <rel name="Contains" num="1">
  "Resource" "NSL-A01-E01"
 </rel>
 <rel name="BitFile">"NewSLBitFile" "NewSLBitFile-Endcap"</rel>
 <rel name="LUTFile">"L1MuESLLUT" "LUTFile-Ver0"</rel>
 <rel name="DelayFPGA">"NewSLDelay" "NewSLDelay-A01-E01"</rel>
 <rel name="MaskFPGA">"NewSLMask" "NewSLMask-A01-E01"</rel>
 <rel name="InnerCoincidenceFlagFPGA">"L1MuESLInnerCoincidenceFlag" "L1MuESLInnerCoincidenceFlag-A01-E01"</rel>
 <rel name="ReadoutFPGA">"NewSLReadout" "NewSLReadout-A01-E01"</rel>
</obj>
```

in muons/segments/TGC/NSL/NewSL_board.data.xml

```
<obj class="NewSLBitFile" id="NewSLBitFile-Endcap">
 <attr name="BitFile" type="string">"L1MuESLEndcapFirmware-02-03-0d.bit"</attr>
 <attr name="BitFileVersion" type="u16">0x23d</attr>
</obj>
```

in muons/segments/TGC/NSL/NewSL_bitfile.data.xml

図 5.12　New SL の OKS オブジェクト (Resource) の定義。赤枠内では、schema ファイルで定義した NewSLBoard class を用いて NewSLBoard-A01-E01 という Resource を data ファイルで定義している。この NewSLBoard-A01-E01 Resource の配下にあるコンフィギュレーションパラメータのオブジェクトも、また別の data ファイルに用意している。ここでは青枠内に、NewSLBitFile-Endcap という FPGA ファームウェアを指定するためのオブジェクトを例として載せている。

5.3.2　Segments & Resources を用いたシステムの階層構造の開発

物理データを収集するためのランでは、図 5.7 に示した Run Control Panel を用いて ATLAS の検出器システムに含まれるすべてのエレクトロニクスの制御を行わなければならない。とあるエレクトロニクスが何らかの問題によって使えない状況になった場合、図 5.8 に示した Segments & Resources からそのエレクトロニクスに対応するオブジェクトを disable しなければならない。もしその使えないエレクトロニクスが他のエレクトロニクスの動作に必要ならば、それらに対応するオブジェクトも自動的に disable されるべきである。Enable する際の操作についても同様のことが言える。堅牢性に長けた統合制御システムを構築するためには、検出器システムの階層構造を適切に

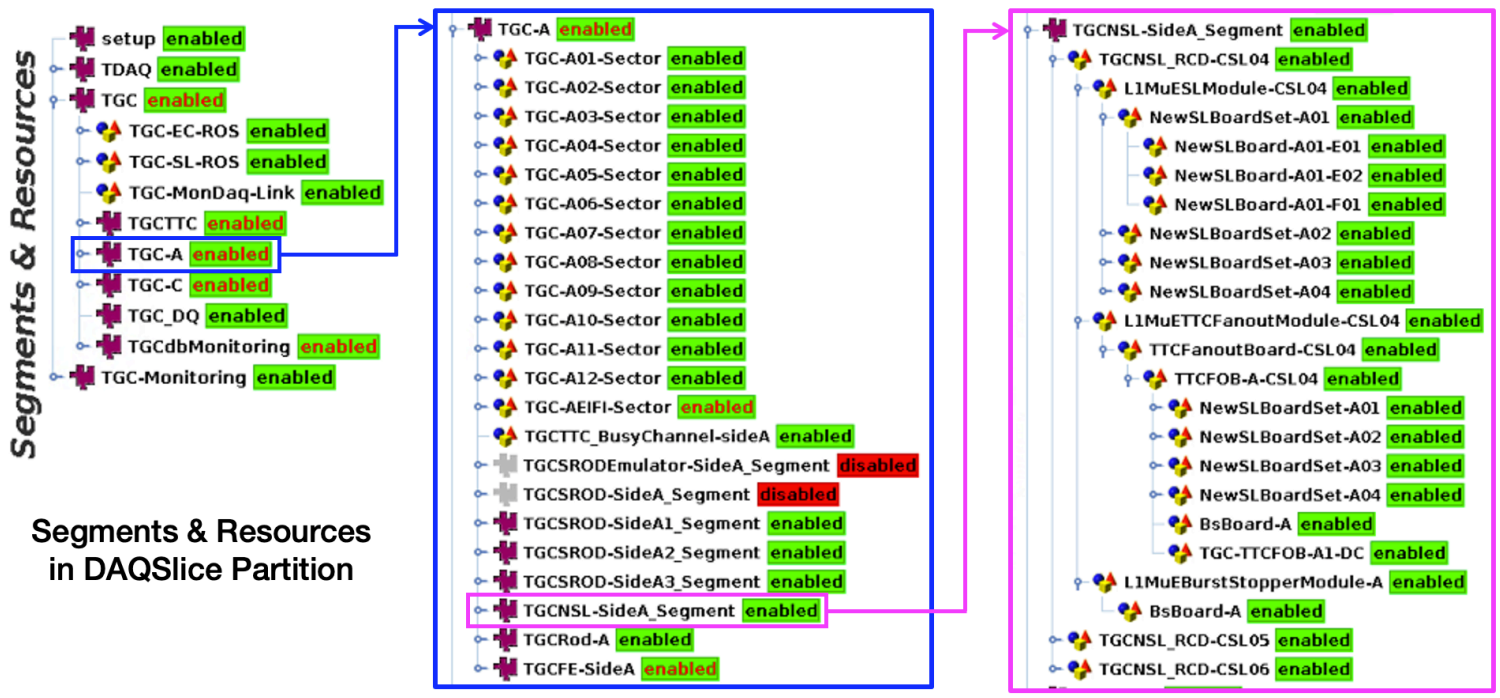


図 5.13　DAQSlice Partition の Segments & Resources と New SL クレートのエレクトロニクスを制御する Segment。DAQSlice Partition に含まれる 4 つの Segment のうち TGC Segment が L1MuE トリガーおよび TGC システムに関係するエレクトロニクスやアプリケーションのオブジェクトを内包する。TGC Segment の配下には A/C-side の L1MuE トリガーと TGC システムを制御する TGC-A/C Segment が含まれる。ここでは代表的に TGC-A の構造を青枠内に示している。本研究では TGC-A/C のそれぞれの配下に A-side と C-side の New SL クレートのすべてのエレクトロニクスを制御するための Segment を新たに追加した。マゼンタ色の枠内にはその代表として、A-side の New SL クレートを司る TGCNSL-SideA_Segment の構造を示している。

組み込み、この enable/disable の伝播を実現する必要がある。そこで本研究では Run 3 に向けて、New SL クレートに対応する Segment を新たに導入したのち、L1MuE トリガーおよび TGC システムの Segments & Resources を全体的に吟味し、再編した。本小節では L1MuE トリガーおよび TGC システムの Segments & Resources の全体像についてまとめたのち、新しい New SL クレートの Segment の開発について重点的に述べる[*4]。

まずは L1MuE トリガーおよび TGC システムの Segments & Resources の全体像について説明する。図 5.13 に主に開発を行う際に使用した TGC の Standalone Partition である DAQSlice Partition (part_TGC_DAQSlice) が含む Segments & Resources を示す。DAQSlice Partition には 4 つの Segment が含まれるが、これらのうち TGC という名前の Segment は L1MuE トリガーおよび TGC に関係するすべてのエレクトロニクスやソフトウェアの Segment や Resource を含む最上位の Segment である。この TGC Segment には ROS に対応する ResourceSet や TGC の TTC システムの Segment、そして A-side および C-side の L1MuE トリガーと TGC のエレクトロニクスを制御する TGC-A と TGC-C という Segment が含まれる。本研究では New SL クレートのすべてのエレクトロニクスの統合制御を実現するべく、図 5.13 の青色の枠に示すように、TGC-A および TGC-C の配下に New SL クレート用の Segment を新たに用意した。

では New SL クレートを制御するために新たに開発した Segment について説明していく。ここでは TGC-A Segment の配下に新たに設けた TGCNSL-SideA_Segment の内部構造を図 5.13 に示した。この直下には TGC-NSL_RCD-CSL0X (X=4,5,6) という名前の ResourceSetAND を用意した。これらは Run Control Driver (RCD) という class のオブジェクトで、A-side の New SL クレートの各 SBC 上で走る Run Control Application に対応する。3 つとも似た構造を持つので、TGCNSL_RCD-CSL04 を例に挙げてその内部構造を説明する。この配下には L1MuESLModule-CSL04、L1MuETTCFanoutModule-CSL04、および L1MuEBurstStopperModule-

[*4] TGC システム全体の Segments & Resources の具体的な吟味と再編については付録 C.3.3 に載せる。

A という 3 種類のエレクトロニクス制御オブジェクトを用意することで、RCD がそれぞれのソフトウェアアプリケーションを走らせられるようにした。L1MuESLModule-CSL04 の配下には 1/12 セクターごとに New SL をまとめた NewSLBoardSet-AXX (XX=01–04) という ResourceSetAND を準備した。それぞれは図 5.12 に示したような New SL ボードを表す Resource を 3 つ (Endcap SL 用 2 つと Forward SL 用 1 つ) 含む。L1MuETTCFanoutModule-CSL04 の配下には TTCFOB を表す TTCFanoutBoard-CSL04 という ResourceSetOR を用意した。TTCFOB は自身が属する New SL クレートの各ボードに TTC 信号を配布するため、TTCFOB が正常に動作しない場合は同じクレートに属する他のボードも正常に動作しないと想定するべきである。このため、TTCFanoutBoard-CSL04 の配下には TTCFOB-A-CSL04 という、自身が属する New SL クレート内のその他のボードをすべて含む ResourceSetAND を用意している。このような階層構造を設けることで、TTCFOB の enable/disable が同一クレート内の New SL や Burst Stopper に伝播されるようにした。L1MuEBurstStopper-A の配下には Burst Stopper ボードをに対応する BsBoard-A という Resource を用意した。なお CSL05 と CSL06 の New SL クレートには Burst Stopper の代わりに G-Link Converter が挿入されているので、Segments & Resources の構造もそのような実装にしている。C-side についても同様の構造を用意することで、全 72 台の New SL を中心とした Run 3 から新たに導入する New SL クレートのすべてのエレクトロニクスを統合制御できるような環境を初めて構築した。これらの新しい L1MuE トリガーの Segment はすべて TGC システムを制御する TGC Segment の配下に設けたので、Run 2 まで使用していた TGC システムとの同期制御も実現されたことになる[*5]。

本節で議論した L1MuE トリガーと TGC の OKS の開発についてまとめる。まずは Run 3 から導入される New SL クレートの新規エレクトロニクスに対応する Resource およびそれらが持つコンフィギュレーションパラメータのオブジェクトをすべて定義した。次にこれらの Resource を用いて統一的な制御を行うために、New SL クレートを司る Segment を TGC Segment の配下に用意し、その配下には各 Resource の enable/disable の伝播を意識した階層構造を持たせた。これらの開発によって、TDAQ ソフトウェアを用いて New SL クレートのすべてのエレクトロニクスを同期制御できるような統合制御システムを構築した。すなわち 5.2.1 節で列挙した、必要な機能の 1. が達成され、TDAQ ソフトウェアを用いて New SL クレートのすべてのエレクトロニクスを同期制御できるようになった。

5.4 Run 3 に向けた State Transition の考案とオンラインソフトウェアへの実装

Run 3 の L1MuE トリガーを正常な状態で走らせるためには、まず図 5.4 に示した Run Control FSM に沿って New SL クレートおよびその周囲のエレクトロニクスを正しい順序でコンフィギュレーションし、運転開始前にシステム全体を同期させる必要がある。具体的には 5.2.1 節の 2. と 3. で述べたように、各エレクトロニクスの TTC 信号との同期や、全 72 台の New SL における周囲の検出器エレクトロニクスとの光リンクの確立を正しい手順で行う必要がある。そこで本研究ではこれらの同期を実現するべく、各エレクトロニクスのソフトウェアアプリケーションにおける Run Control の状態遷移 (state transition) を吟味し、Run 3 に向けた state transition を新たに実装した。具体的には 5.2.3 節で説明した New SL クレートの制御に必要なパッケージ群、TGCFEModule、および TGCDelayModule を開発した。

State transition を吟味する上で最も重要なことは、運転状態に遷移させるための RC Command である CONFIGURE、CONNECT、START の際にどのような操作を行うかである。そこで図 5.14 に、Run 3 に向けて新たに考案した state transition の CONFIGURE、CONNECT、START における運転操作手順を示す。New SL ク

[*5] より正確には、Run 2 の TGC システムとの同期制御を実現するために TGC Segment 全体の吟味と再編を行なった。しかしその詳細はかなりテクニカルなので付録 C.3.3 に記す。

RC Command	SubTransition	New SL クレート	TGC のフロントエンド (CCI クレート)	TGC の TTCシステム (TTC クレート)	他のサブシステム (NSW, RPC BIS78, TMDB, MUCTPI)
CONFIGURE		OKS からパラメータを取得	OKS からパラメータを取得	OKS からパラメータを取得	OKS からパラメータを取得
	TGCTTC_INIT	・ファームウェアの確認、書き込み ・TTCFOB の TTCrx のリセット	TTCrx のリセット	Delay Module から TTCrx のリセットコマンドを送信	
	TGCTTC_CONFIG	TTCFOB の TTCrx の設定	TTCrx の設定		
		・Delay や mask など、OKSから取得したパラメータの設定 ・New SL、Converter で GTX TX のレジスタとロジックのリセット	・HPT、EI/FI PS Board の G-Link を idle mode に設定 ・PS Board、SSW、HPT のレジスタの設定	主に TTCvi の設定	・TMDB の G-Link を idle mode に設定 ・NSW、RPC BIS78 で光シリアル通信のトランスミッターのレジスタとロジックのリセット
CONNECT		・New SL で HPT、TMDB との G-Link 通信の確立 ・Converter で EI/FI PS Board との G-Link 通信の確立 ・NSW、RPC BIS78 との通信用に GTX RX のレジスタのリセット		主に LTP の設定	MUCTPI で光シリアル通信のレシーバーのレジスタのリセット
START	TGCFESL_FIFO_RESET	FIFO のリセット	FIFO のリセット		
			HPT、EI/FI PS Board の G-Link を data mode に設定		TMDB の G-Link を data mode に設定

図 5.14　Run 3 の state transition の全体像。CONFIGURE、CONNECT、START で行う重要な操作を抜粋している。

レート、TGC のフロントエンド、および TGC の TTC システムで行う重要な操作、そして他の検出器システムで行う New SL との通信を確立するための操作を抜粋している。これを考案するにあたって、5.2.1 節の 2. と 3. を達成すべく次の 2 点に重きを置いた：

- 各エレクトロニクスの安定動作を保証するために、最優先でファームウェアや TTC 関係の設定を行う
- システム間の通信を安定して確立するために、その順序は明確に定義して行う

以下に各 RC Command で各システムでどのような操作を行うかについて説明する。

まず CONFIGURE では New SL クレート、TGC のフロントエンド、TGC の TTC システム間で同期した形で TTC 関係の設定を行わなければならないため、2 つの “SubTransition” を用意して段階的な処理を実装した。SubTransition とは OKS にて Segment 単位で用意できる処理段階で、ATLAS の RC Command に応じてその Segment の配下にあるエレクトロニクスを同期制御するために用いられる。今回新たに提案した state transition では、OKS パラメータを取得したのち、システム全体で TTCrx をリセットするための SubTransition として TGCTTC_INIT を用意した。このとき TTC クレートにある Delay Module によって TTCrx をリセットするためのブロードキャストコマンドが配布され、New SL クレート内の TTCFOB 上および TGC のフロントエンドにある TTCrx がすべてリセットされるようにした。並行して、New SL クレート内の各エレクトロニクスの FPGA のファームウェアの確認および必要に応じた書き込み操作も行われる。次の SubTransition である TGCTTC_CONFIG では、リセットした TTCrx に OKS から取得したパラメータを書き込むことで、すべてのエレクトロニクスが正しいタイミングで TTC 信号を受け取ること保証する。こうして SubTransition を積極的に用いて、同期した形でファームウェアや TTC の設定を行われるように実装し、システム全体の安定動作を保証した。

TGCTTC_CONFIG の直後はその他のコンフィギュレーションに加え、システム間の通信を確立するための送信側で準備を行うようにした。New SL では MUCTPI にトリガー判定情報を、G-Link Converter では New SL に TGC EI/FI の信号を送信するために GTX TX のロジックとレジスタのリセットを行う。TGC のフロントエンドでは HPT および EI/FI 用の PS Board にて、New SL にデータを送るために G-Link のステータスを idle mode に設定する。TMDB でも同じタイミングで G-Link を idle mode にしてもらう。また NSW や RPC BIS78 のエレクトロニクスでは、New SL にトリガーデータを送信するために光シリアル通信のトランスミッターのロジックとレジスタのリセットを行なってもらう。

次の RC Command である CONNECT ではデータの受信側で通信を確立するための設定を行うようにした。

New SL では HPT と TMDB との G-Link 通信を確立するための操作、および NSW と RPC BIS78 との光シリアル通信を確立するために GTX RX のレジスタのリセットを行う。また G-Link Converter では EI/FI 用の PS Board との G-Link 通信を確立するための操作を行う。MUCTPI では New SL からのトリガー判定情報を受信するために光シリアル通信のレシーバーのレジスタのリセットをしてもらう。

START では TGCFESL_FIFO_RESET という SubTransition を用意して、新しいデータの送受信を開始する前に古いデータをクリアすべく、New SL クレート内および TGC フロントエンドの各エレクトロニクスの FIFO をリセットするように実装した。また G-Link 通信ではデータ送信を始める前に送信側のステータスを data mode にする必要があるので、TGCFESL_FIFO_RESET の直後に HPT、EI/FI 用の PS Board、および TMDB でそのようにする。このようにして各システムで行う操作の順序をあらわに定義することで、New SL と各検出器システム間の通信が安定して確立されるような実装を行なった。

すなわち本研究では、各エレクトロニクスの TTC 信号との同期を優先的に行い、光リンクの確立の手順を ATLAS の Run Control FSM に沿って明確な順序で行うような state transition を考案し、それを L1MuE トリガーの新規エレクトロニクス、TGC のフロントエンドエレクトロニクス、そして TGC の TTC システムを制御するソフトウェアアプリケーションに実装した。この開発研究によって 5.2.1 節で述べた、要求される機能の 2. と 3. が達成され、L1MuE トリガーシステムが TTC 信号と同期してトリガー信号を出力できるようになった。

TTC Restart

物理データを収集している際に何らかの理由によって検出器システムの制御アプリケーションがクラッシュして再起不能となった場合、busy 信号を出力してデータ収集を止め、そのシステム全体を復帰させる手続きが必要となる。この操作は TTC Restart と呼ばれ、Run Control Panel を用いて実行する。TTC Restart を行なっている最中も LHC の陽子ビームが出ているので、復帰にかかる時間が長いほど物理データを失うことになる。よって TTC Restart ではなるべく必要のない操作を省き、復帰にかかる時間を短くしなければならない。また TTC Restart は必ずしも ATLAS 検出器全体で行うものではなく、クラッシュした検出器システムのみに対して実行する場合が多い。すなわち正常に走っている他の検出器システムとの同期を保ったまま、該当する検出器システムのみを復帰させることが要求される。本研究ではこれらの事項を念頭に置いて L1MuE トリガーに TTC Restart 機能を実装し、5.2.1 節の 4. で説明した素早い reconfiguration を可能にした。

New SL クレートを中心とした Run 3 の L1MuE トリガーに対して TTC Restart を実行するには、OKS の階層構造の関係上、図 5.13 にも示した TGC Segment に対して行うことになる。すなわち TGC Segment の外にあるエレクトロニクス (i.e. MUCTPI、TMDB、NSW、RPC BIS78) は reconfiguration されないので、これらと確立した通信はリセットしてはならない。また、前述したように TTC Restart の所要時間はなるべく短くするべきである。これらの 2 つの事項に基づき、通常の configuration の手順から、一部の操作を差し引く形で、New SL クレートのエレクトロニクスのソフトウェアアプリケーションに TTC Restart の機能を実装した[*6]。

では TTC Restart の際に飛ばすべき具体的な操作について説明する。ここでは New SL で行う操作を例に挙げる。まず故意的な場合を除いて、ラン中に New SL の FPGA ファームウェアが書き換わることはないはずなので、ファームウェアのバージョン確認や書き込みの操作は飛ばすようにした。また同様の理由によって LUT の確認も飛ばす[*7]。また TGC Segment 外のシステムとの通信に関係するリセット操作はすべて飛ばす。MUCTPI との通信のために CONFIGURE で行う GTX TX のロジックとレジスタのリセット、および CONNECT で行う TMDB との G-Link 通信を確立するための操作や NSW と RPC BIS78 との通信のために行う GTX RX のレジスタのリセットはすべて飛ばすように定義した。New SL クレートの他のエレクトロニクスのソフトウェアにも同様の実装を行なった。このようにして他のシステムとの通信を確立した状態を保ちながら、L1MuE トリガーおよび TGC システムを素早く復帰させる手順を構築した。この TTC Restart 機能の実装によって、5.2.1 節の 4. が達成された。

---

*6 具体的な実装方法は付録 D.1 に示す。

*7 現在実装する予定の LUT の確認の操作には数分程度要することが分かっているので、失う物理データを少なくできるという点から、この操作を飛ばすことのメリットは大きいと考えられる。

5.5 モニタリング機能の開発

5.1.3 節で述べたように、検出器エレクトロニクスの誤動作またはその兆候を検知・防止するために、システム全体の行き渡ったモニタリングを実現する必要がある。本研究では Run 3 の新規エレクトロニクスの TTCrx と SFP+ 光トランシーバと I^2C 通信を行うための API を開発し、これらのモジュールの状態を IS 経由でモニタリングする機能を OKS を含むオンラインソフトウェアに実装した。加えて、Grafana を使って IS の情報をウェブ上でオンラインモニタリングする機能を TGC の Standalone Partition に追加した。これらの開発によって、5.2.1 節に挙げた要求機能の 5. が達成され、より優れた堅牢性を持つ統合制御システムが実現された。本節ではこれらのモニタリング機能の開発について説明する。

5.5.1 I^2C 通信の概要

TTCFOB 上の TTCrx や、New SL と G-Link Converter に挿す SFP+ 光トランシーバのレジスタとの通信には I^2C[*8] というシリアルバス通信規格が採用されている。また、これらのモジュールとの通信は VME 経由で FPGA または CPLD のレジスタをビットバンギングすることで行えるように設計されている。すなわちオンラインソフトウェアを用いてこれらのモジュールを操作するには、I^2C 規格の手順に従って FPGA または CPLD のレジスタを操作する API が必要となる。本小節ではモニタリング機能の開発の導入として、I^2C の通信方式について、TTCrx と SFP+ 光トランシーバに共通する部分を説明する。

一般的な I^2C 通信では、マスターがシリアルデータ (SDA) 線とシリアルクロック (SCL) 線という 2 本の接続線を制御し、複数のスレーブとの通信を行う。その通信方式の模式図を図 5.15 に示す。今回扱う I^2C 通信では FPGA または CPLD がマスターの役割を果たし、2 つのスレーブと通信する。その通信手順について以下に簡単にまとめる[*9]：

1. マスターの読み書きが可能なレジスタ (SDA_W と SCL_W) を用いて SDA 線と SCL 線の high (1) と low (0) を制御し、通信したいスレーブのアドレスを指定する。
2. 該当するスレーブが ACK という応答信号を返すので、それをマスター側の read-only レジスタ (SDA_R) で読む。これによってデータ通信が正しく行われているかを確認できる。
3. 1. と 2. を繰り返し、マスターを操作して 8 ビット単位でスレーブとデータ通信を行う。

本研究で扱った TTCrx と SFP+ 光トランシーバはともに I^2C 通信規格を採用しているものの、それぞれでスレーブの扱いが異なる。すなわちこれらのモジュールとの通信を実現するためには、それぞれに適合した通信用 API を別々に開発する必要があった。それらの詳細は次の小節以降で説明する。

5.5.2 TTCrx のコンフィギューレーションとモニタリング機能の開発

TTCFOB 上の TTCrx は TGC の TTC システムから受け取った TTC 信号に適切な遅延をかけるためのチップである。受信した TTC 信号は L1MuE トリガーの最終的なトリガー判定を担う New SL に配られる。すなわち L1MuE トリガーシステムの安定動作を保証するためには、TTCFOB 上の TTCrx が正しくコンフィギュレーションされ、運転中も正常に動作している必要がある。そこで本研究では TTCrx のレジスタを読み書きするための API を開発し、そのコンフィギュレーションとモニタリングを行う機能を TTCFOB のソフトウェアアプリケーションに実装した。

---

*8 Inter-Integrated Circuit の略。

*9 より具体的な通信手順は付録 D.2 に載せる。

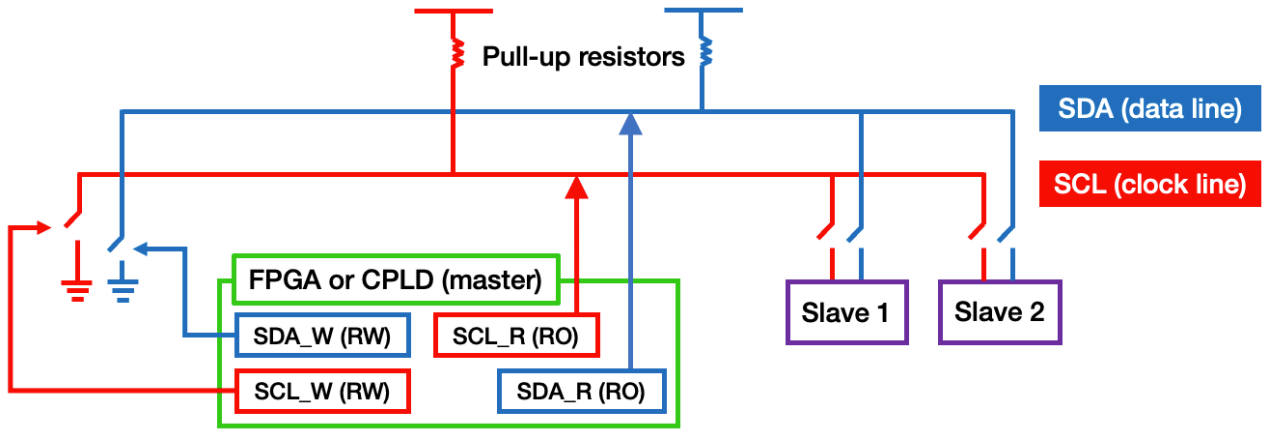


図 5.15　I^{2}C の通信方式の模式図。TTCFOB では FPGA が、New SL と G-Link Converter では CPLD がマスターの役割を果たし、2 つのスレーブと通信する。信号はすべてオープンドレイン方式で取り扱われる。

TTCrx には計 20 個のレジスタがある。表 5.4 にその代表的なレジスタを挙げる。これらの読み書きは I^{2}C 規格に基づいて行う。とあるレジスタの読み書きには、使用している TTCrx が持つ固有の 6 ビットの I2C_ID と、該当するレジスタの I^{2}C アドレスが必要になる[*10]。その通信手順について簡単にまとめる。まず TTCrx は I^{2}C ポインタと I^{2}C データという、2 つのレジスタをスレーブとして持つ。これらのスレーブのアドレスは表 5.5 に示すように、TTCrx が持つ 6 ビットの I2C_ID から計算される。この 2 つのスレーブを用いた操作は以下の通りである：

1. まずは I^{2}C ポインタに通信したいレジスタの I^{2}C アドレス (e.g. Status レジスタなら 22) を書き込む。具体的には、通信先の TTCrx が持つ I^{2}C ポインタの 7 ビットのスレーブアドレスを SDA に送る。最後の 1 ビットは書き込みを意味する 0 を送る。その後に、通信したいレジスタの I^{2}C アドレスを 8 ビットの情報として送る。
2. 1. で指定したレジスタが保有する値が I^{2}C データの値として反映される。
3. I^{2}C データの 7 ビットのスレーブアドレスと、読み出し/書き込みを意味する 1/0 の計 8 ビット情報を SDA に送る。読み出しの場合はスレーブから SDA に 8 ビットの情報が順次送られてくる。書き込みの場合はマスターから SDA に 8 ビットの情報を送る。

上記の手順に従って TTCFOB 上の TTCrx の全 20 個のレジスタと通信を行うために、図 5.11 にも示した L1MuEVmeApi パッケージの下に 2 種類の API を開発した。片方は SDA における 8 ビット単位の読み書きやスレーブから来る ACK 信号の確認など、I^{2}C の基本的な操作を VME および TTCFOB 上の FPGA を経由して行うための API (TgcI2cApi) である。もう片方は TgcI2cApi を用いて、TTCrx が採用している I^{2}C のポインタレジスタとデータレジスタのロジックを用いて読み書きを実現するための API (TgcTtcrxApi) である。TgcTtcrxApi では TTCrx のレジスタリストを作成することで、各レジスタの名称、アドレス、初期値などをまとめて管理できるようにした。

最後に、TTCFOB を制御する L1MuETTCFanoutModule パッケージに TTCrx のコンフィギュレーションとモニタリングを行う機能を実装し、TDAQ ソフトウェアおよび Partition を用いた制御を実現した。コンフィギュレーションは表 5.4 に示した 3 つのタイミング調整を担うレジスタと Control レジスタに対して行うようにした。コンフィギュレーションでは図 5.14 に示したように、CONFIGURE 時に OKS から取得したパラメータを前述した 4 つのレジスタに書き込む。さらにそれらの値を読み出すことで、実際に正しい値が書き込めているかの確認もするようにした。モニタリングは運転中に値が変動すべきでない 15 個のレジスタに対して行うようにした。RC State が RUNNING のときに、OKS で設定した時間間隔[*11]で 15 個のレジスタの値を読み出して図 5.16 に示すような

[*10] 6 ビットの I2C_ID がわからない場合は TTCrx との通信が一切できない。そこで本研究ではそれを特定する方法も開発した。詳細は付録 D.3 で説明する。
[*11] 現在の設定では 10 秒間隔。

表 5.4　TTCrx の代表的なレジスタ [37]。それぞれ 8 ビットの値を保有する。

レジスタ名	I^2C アドレス	概要
Fine Delay 1	0	TTC 信号の遅延時間を設定する
Fine Delay 2	1	
Coarse Delay	2	
Control	3	出力するクロックの種類などを設定する
MMB[1:0], I2C_ID[5:0]	18	I2C_ID は I^2C 通信を行うために必要
Status	22	TTCrx のチップの状態を表す

表 5.5　TTCrx の I^2C ポインタレジスタとデータレジスタ代表的なレジスタ [37]。

I^2C のスレーブ名	7 ビットのスレーブアドレス
ポインタレジスタ	I2C_ID[5:0]×2
データレジスタ	I2C_ID[5:0]×2 + 1

```
<obj class="TTCFanoutTTCrxRegisters" id="TTCFanoutTTCrxRegisters-CSL01">
 <attr name="FineDelay1"  type="u8">0x0</attr>
 <attr name="FineDelay2"  type="u8">0x0</attr>
 <attr name="CoarseDelay"  type="u8">0x22</attr>
 <attr name="Control"  type="u8">0xb3</attr>
 <attr name="SingleErrorCount0-7"  type="u8">0x0</attr>
 <attr name="SingleErrorCount8-15"  type="u8">0x0</attr>
 <attr name="DoubleErrorCount0-7"  type="u8">0x0</attr>
 <attr name="SeuErrorCount8-15"  type="u8">0x0</attr>
 <attr name="Id0-7"  type="u8">0x6c</attr>
 <attr name="MasterModeA0-1"  type="u8">0x0</attr>
 <attr name="MasterModeB0-1_I2c_Id0-5"  type="u8">0x2c</attr>
 <attr name="Config1"  type="u8">0x1a</attr>
 <attr name="Config2"  type="u8">0x84</attr>
 <attr name="Config3"  type="u8">0xa7</antr>
 <attr name="Status"  type="u8">0xe0</attr>
</obj>
```

図 5.16　TTCFOB の TTCrx のレジスタの OKS 設定値の一部。運転中に読み出した値をこれらの値と比較することで、正しい値が常に書き込まれているかをオンラインモニタリングする。

OKS の値と照らし合わせ、値が異なる場合は ERS を用いて WARNING メッセージを出すように実装した。またそれらの読み出した値を IS サーバーに送信し、オンラインモニタリングできるようにした。このようにして ERS と IS を用いて二重でモニタリングできるようにすることで、全 6 台の TTCFOB 上の TTCrx の誤動作やその兆候を素早く検知できるようなモニタリングシステムを構築した。

5.5.3　SFP+ 光トランシーバのモニタリング機能の開発

New SL や G-Link Converter に挿入する SFP+ 光トランシーバ [38] は、光信号を送受信するために使用するモジュールである。これらが正常に動作しない場合、8b/10b 規格で送られてくる TGC EI/FI、NSW、および RPC BIS78 のヒット信号が正しく受信されない可能性がある。すなわち SFP+ 光トランシーバの動作状況をオンラインモニタリングすることでデータの送受信が安定して行われていることを保証し、検出器システムの堅牢性を向上させることが重要である。

SFP+ 光トランシーバが持つレジスタとの通信にも 5.5.1 節で説明した I^2C 規格が採用されている。ただし、スレーブの使い方は前節で説明した TTCrx とは異なる。SFP+ はアドレスとして A0h (0xa0) と A2h (0xa2) を持

つ 2 つの “デバイス” をスレーブとして持っており、それぞれの配下に複数のレジスタが存在する。今回使用する AVAGO 社の AFBR-709SMZ [39] の場合は、A0h の配下に製品情報などの固定データが入った read-only レジスタが、A2h の配下には温度や供給電圧などの情報が入ったモニタリング用のレジスタがある。これらの各レジスタとの通信は以下の手順で行う：

1. 通信したいレジスタを持つスレーブの 7 ビットのデバイスアドレス (2b’101000X) と、読み出し/書き込みを意味する 1/0 の計 8 ビットの情報を SDA にて送る。
2. 通信したいレジスタの 8 ビットのレジスタアドレスを SDA にて送る。
3. 8 ビットの値を読み出す、あるいは書き込む。

まずは L1MuEVmeApi パッケージの配下にある New SL と G-Link Converter の VME インターフェース (TGC_NSLVmeInterface と TGC_CNVVmeInterface) に適切な関数を実装することで、上記の基本的な読み書きの操作を VME そして CPLD 経由で行えるようにした。

A2h の配下のレジスタを用いてモニタリングできるパラメータを表 5.6 に示す。これらの値は 16 ビットで表記され、それぞれに 2 つの 8 ビットレジスタが割り当てられている。またこれらの 5 つのパラメータに対して warning および alarm を出すための閾値を、高低のそれぞれについて設定できる。実際の値がそれらの閾値を超えた場合、4 つある “フラグビット” レジスタの対応するビットが立つような仕掛けになっている。すなわちモニタリングの対象となるのは 5 つのパラメータの値を保有する 10 個のレジスタ、4 個のフラグビットレジスタ、そしてその他の情報を持つ Status/Control レジスタの計 15 個のレジスタである。

表 5.6　SFP+ 光トランシーバのモニタリングできるパラメータ。それぞれに 16 ビットが割り当てられている。温度のみ符号付きである。

パラメータ名	最下位ビット相当量 d	概要
Temp	1/256 °C	温度。符号付き。2 の補数表記を採用。
Vcc	100 μV	供給電圧。符号なし。
Tx Bias	2 μA	送信側のレーザーバイアス電流。符号なし。
Tx Pwr	0.1 μW	送信した平均の光量。符号なし。
Rx Pwr	0.1 μW	受信した平均の光量。符号なし。

そこで次に、モニタリング対象であるレジスタの管理や、読み出したパラメータのレジスタの値のデコードを行う SFP+ 光トランシーバの API (TgcSfpApi) を開発した。前者についてはレジスタの名称、アドレス、デフォルト値をリストして管理し、New SL や G-Link Converter のソフトウェアアプリケーションでそれらを参照できるようにした。また後者については、2 つの 8 ビットレジスタから読み出した 16 ビットの値を、表 5.6 の最下位ビット相当量 d を用いて実際の値に変換できるようにした。この変換後の値を IS に送信することで、よりユーザーフレンドリーなオンラインモニタリングを実現した。

最後に、New SL を制御する L1MuESLModule と G-Link Converter を制御する L1MuEGLinkCnvModule に、TgcSfpApi のレジスタリストやデコード機能を用いて運転中に SFP+ 光トランシーバのモニタリングを行う機能を搭載した。しかし 2021 年 1 月現在、USA15 に NSW や RPC BIS78 のエレクトロニクスはまだ設置されていないため、New SL のほとんどの SFP+ 光トランシーバは光信号を受信できる環境になく、立ちっぱなしのフラグビットもある。そのためモニタリングによる ERS メッセージの発行は現時点では無効にしており、IS に値を投げるという形でのみオンラインモニタリングを行なっている。Run 3 の開始までには、フラグビットレジスタや Status/Control レジスタの値に応じて、適切に ERS メッセージを発行するような実装をする予定である。

本小節で説明した SFP+ 光トランシーバのモニタリング機能の開発によって、全 72 台の New SL および全 4 台の G-Link Converter の合計 576 個の SFP+ 光トランシーバを用いた 8b/10b 規格の光通信の状況を常に把握できるようになった。すなわち通信の安定性を保証するという点で、L1MuE トリガーシステムの堅牢性を大幅に向上さ

```
<obj class="Partition" id="part_TGC_DAQSlice">
 <attr name="IPCRef" type="string" val="$(TDAQ_IPC_INIT_REF)"/>
 <attr name="DBPath" type="string" val="$(TDAQ_DB_PATH)"/>
 <attr name="DBName" type="string" val="$(TDAQ_DB_DATA)"/>
 <attr name="DBVersion" type="string" val="$(TDAQ_DB_VERSION)"/>
                ⋮
 <rel name="DataFlowParameters" class="DFParameters" id="DFParameters"/>
 <rel name="IS_InformationSource" class="IS_InformationSources" id="EventsAndRates"/>
 <rel name="MasterTrigger" class="MasterTrigger" id="MasterTrigger_DAQSlice"/>
 <rel name="TriggerConfiguration" class="TriggerConfiguration" id="TrigConf_TGCDAQSlice"/>
 <rel name="OnlineInfrastructureApplications">
  <ref class="PBeastApplication" id="pbeast-receiver"/>
 </rel>
</obj>
```

in muons/partitions/part_TGC_DAQSlice.data.xml

図 5.17　DAQSlice Partition における PBEAST モニタリングの実装。赤枠で囲っている部分のように “pbeast-receiver” を Partition オブジェクトの OnlineInfrastructureApplications と紐付けることで、IS へ送信した情報が PBEAST サーバーにも送られるようになる。

せることに成功した。SFP+ 光トランシーバの具体的なモニタリングの手法については次の小節で述べる。

5.5.4　Grafana を用いたオンラインモニタリングの実現

5.1.3 節でも述べたように、IS に送信された情報は瞬間的にしか保持されないため、その情報を永続的に保存する PBEAST というモニタリングサービスが開発された。PBEAST に送信された情報は Grafana というデータ可視化ツールを用いてウェブ上でモニタリングすることができるようになっている。Grafana を用いたモニタリングを可能にすることで、TDAQ ソフトウェアを用いた統合制御運転によって取得した検出器システムの情報がユーザーフレンドリーな形式で、オンラインモニタリングできるようになる。そこで本研究では Standalone Partition を用いた試運転中に得た、5.5.2 節や 5.5.3 節で説明した L1MuE トリガーの新規エレクトロニクスなどの情報を Grafana にてモニタリングできるようにした。これによって 5.2.1 節で列挙した L1MuE トリガーの統合制御に要求される機能の 5. が達成されることになる。

Standalone Partition で IS の情報を PBEAST サーバーに送信するためには、図 5.17 に示すように、その Partition オブジェクトに pbeast-receiver というアプリケーションを紐づける必要がある。加えて PBEAST サーバー側でその Partition の情報を受信するための設定を行うことで、PBEAST を用いたモニタリングのパスが開通される。今回は L1MuE トリガーや TGC のコミッショニングで頻繁に使う DAQSlice Partition において PBEAST を用いたモニタリングパスを開通させた。

PBEAST サーバーに送られた情報はすべて Grafana を用いて、ウェブ上でオンラインモニタリングできる。Grafana でのモニタリングは Dashboard を用いて行う。図 5.18 に実際に作成した Dashboard のスクリーンショットを示す。Dashboard 上には複数の Panel を配置することで、複数のプロットを同時にモニタリングすることができる。各 Panel のプロットを表示するには、図 5.19 に示すように PBEAST の Query を指定する必要がある。Query は Partition の名前、IS のタイプ、IS パラメータの名前、そして IS のパスから構成される。これらをすべて正しく指定すると該当するデータが PBEAST から取得され、表示される。また、複数の Panel は Row を用いてまとめることができ、Row ごとに Panel の表示と非表示が簡単に切り替えられるようになっている。さらに Dashboard に Variable を追加することで、変数を用いてプロットの表示の仕方を変えることもできる。

図 5.18 に示した Dashboard では Partition 名を Variable として選べるように実装し、それぞれの Partition で取得したデータを瞬時に見れるようにした。この図の特に右上のプロットは受信した光量を表すもので、G-Link Converter からデータを受信する SFP+ のみが 0 でない値を取っている。これは 2021 年 1 月現在、G-Link Converter から光信号を受信する SFP+ のみに光ファイバーが挿入されているからである。SFP+ 光トランシーバ

図 5.18　本研究で開発した Grafana の Dashboard。ここでは DAQSlice Partition を用いて取得した表 5.6 の 5 つのパラメータの時間変化をまとめて表示している。この図は DAQSlice Partition を用いて取得した、A05-E09 の New SL に挿入されている 3 つの SFP+ 光トランシーバのパラメータの時間変化を表している。赤い点が G-Link Converter からデータ受信する SFP+ を、青い点が RPC BIS78 の Pad Board からデータを受信する SFP+ を、紫の点が NSW TP からデータを受信する SFP+ を表している。

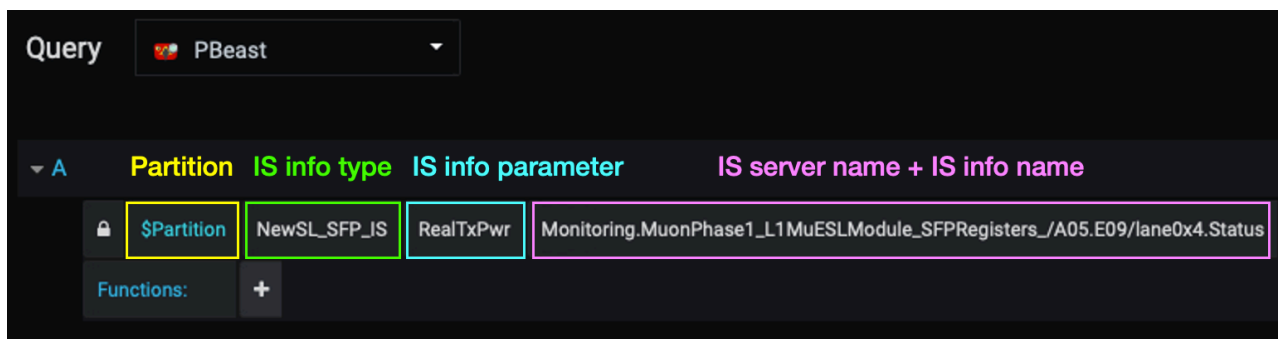


図 5.19　Grafana の Query。Partition の名前は Variable として設定しているので、それを引用するようにしている。

以外にも、5.5.2 節で説明した TTCFOB 上の TTCrx のパラメータや、6 章で説明する SROD の読み出しデータを処理する ROS の状態など、新しい L1MuE トリガーシステムのモニタリングを可能にした。すなわち、5.2.1 節で挙げた L1MuE トリガーシステムの行き渡ったモニタリング機能の実装はほぼ達成された。監視したい重要なパラメータは随時検討し、Grafana を用いてオンラインモニタリングできるようにしていく。

また、上記のように Grafana を用いたオンラインモニタリングを活用することで、現地から離れていても Run 3 の新規エレクトロニクスの状況を把握することができるようになった。すなわち Run 3 に向けたコミッショニングを行う上で、Standalone Partition を用いたオンラインモニタリングの実現は重要であると結論付けられる。また Run 3 の物理データを収集するときも、ATLAS Partition で取得した検出器システムのデータは Grafana を用いてオンラインモニタリングする予定である。すなわち Grafana を用いたオンラインモニタリングに関する知見を今のうちに蓄えておくことで、Run 3 の開始までに充実したモニタリングシステムを開発することができる。本研究で

開始した Grafana を用いた TGC Standalone Partition のオンラインモニタリングを皮切りに、L1MuE トリガーおよび TGC システムの洗練されたモニタリングシステムを構築していく。

5.6 新たに開発した統合制御システムの現状と今後

5.3 節から 5.5 節までで説明したように、本研究では OKS データベースとソフトウェアアプリケーションの開発を通して L1MuE トリガーおよび TGC の検出器システムの統合制御を実現した。これらの開発によって 5.2.1 節で述べた、その統合制御に要求される 5 つの機能を実装・達成した。本節では新たに開発した統合制御システムの現状と今後について簡単にまとめる。

統合制御システムのバリデーション

本研究では TGC の Standalone Partition (主に DAQSlice Partition) を用いて、L1MuE トリガーと TGC の OKS とソフトウェアアプリケーションを開発してきた。しかし実際に物理データを収集するために使用する Partition は、ATLAS のすべての検出器システムを同期制御することができる ATLAS Partition である。すなわち今回構築した統合制御システムが正しく動作するかを検証するためには、ATLAS Partition に本制御システムを組み込み、試運転を行う必要がある。

ATLAS 実験では、各検出器グループが開発してきた制御システムを ATLAS Partition を用いて動作検証するために、“Milestone Week” というキャンペーンが定期的に開催されている。Run 3 に向けた第 1 回の Milestone Week は 2020 年 5 月 18 日から 22 日までの期間に行われた。この第 1 回に向けて、DAQSlice Partition で開発した Segments & Resources を ATLAS Partition に実装した。そしてこの期間中、本研究で開発した L1MuE トリガーおよび TGC の統合制御システムは ATLAS Partition でも正常に動作することが確認できた[*12]。さらに 2020 年では計 4 回の Milestone Week が開催され、そのすべてで本制御システムが問題なく動作することも確認できている。このような形で、本研究で開発してきた統合制御システムのバリデーションは十分に行われている。

統合制御システムを用いたコミッショニング

2021 年 1 月現在、Run 3 の開始に向けて New SL クレートのエレクトロニクスを中心としたコミッショニングが進行している。New SL は複数の検出器エレクトロニクスからヒット信号を受信するが、それらの信号をラッチするときのクロックの位相合わせ、各信号の入力タイミングを揃えるためにかける遅延時間の調節、そして L1MuE トリガーのタイミング (BCID) を ATLAS 検出器のトリガーを司る CTP と合わせる作業などが必要になる。現在は、New SL に信号を送る準備がすでにできている TGC BW と TGC EI/FI のクロックの位相合わせや遅延時間の調節の作業が進行している。これらのコミッショニングでは、本研究で開発した統合制御システムが使われている。

今後の課題

NSW TP、RPC BIS78 Pad Board、MUCTPI などの Run 3 の新規エレクトロニクスはまだ USA15 に設置されていないため、実際に物理データの収集を行う環境では、これらと New SL との通信試験はまだ実施されていない。また TMDB はすでに USA15 に設置されているが、まだ Run Control Panel を使わない形で通信安定性の試験が進行している。すなわち TGC を除く検出器システムとの同期制御の検証はまだ行われていない状況である。これらの検出器エレクトロニクスが USA15 に配置され、それらとの通信安定性が保証されたのち、TDAQ ソフトウェアを用いた同期制御試験を随時行なっていく必要がある。

加えてもう 1 つの課題は、オンラインモニタリングの充実である。本研究では初めて TGC の Standalone Partition で Grafana を用いたオンラインモニタリングを可能にし、SFP+ 光トランシーバや TTCrx をはじめとす

[*12] ただし、New SL の周囲の新規エレクトロニクス (e.g. MUCTPI、NSW TP、RPC BIS78 Pad Board) は USA15 にまだ設置されておらず、TGC 検出器自体の電源も上げていないため、New SL クレートおよび TGC のエレクトロニクスで閉じた形で動作検証がされたことになる。

る一部モジュールのモニタリング機能を開発した。今後はモニタリングすべきパラメータを吟味し、ソフトウェアアプリケーションや OKS に適切な実装を加えていく必要がある。また Grafana の Dashboard を充実させ、優れたユーザーインターフェースを持つモニタリングシステムを Run 3 までに構築していかねばならない。

第 6 章

Level-1 ミューオンエンドキャップトリガーの高速読み出しシステムの開発

Run 3 の Level-1 ミューオンエンドキャップトリガーシステムでは、新たに開発された New SL が複数の検出器エレクトロニクスから来るヒット信号を取りまとめ、最終的なトリガー判定を行う。受信したヒット情報 (トリガー論理回路への入力情報) とトリガー判定出力情報はすべて New SL の FPGA 上に実装される L1 buffer に一時的に保存され、L1A 信号が来たときに対応するバンチ交差のヒット情報およびトリガー判定情報が読み出される。すなわち L1A 信号が来る頻度である 100 kHz で、New SL のトリガー読み出しデータを L1ID ごとにまとめ、ROS に送るような読み出しシステムが必要である。4.2.4 節で述べたように、Run 3 では Software-based ROD (SROD) というソフトウェアベースの読み出しシステムを新たに開発し、トリガーデータの読み出しを行う。このトリガーデータの読み出しは、New SL におけるトリガー論理回路のバリデーションや性能評価を可能にするために必要不可欠な機能である。

2020 年 6 月、ATLAS 回路室である USA15 に設置された SROD 用の PC の設定が完了し、実際に読み出しソフトウェアを走らせることが可能になった。本研究では SROD の OKS とソフトウェアアプリケーションを開発することで、実際に物理データを収集する環境で、上記の高速トリガーデータ読み出しシステムを運転・制御できるようにした。本章ではこの開発研究について議論する。

6.1 Software-based ROD (SROD) の導入

Software-based ROD (SROD) は Run 3 の Level-1 ミューオンエンドキャップトリガーのデータ読み出しシステムの中核をなす。本節では SROD の要求性能およびそのソフトウェアの構造について説明する。

6.1.1 トリガーデータ読み出しパスの全体像と SROD ソフトウェアの要求性能

図 6.1 に Run 3 のトリガーデータ読み出しシステムの全体像を示す。New SL は L1A を受けると、各検出器エレクトロニクスから来たヒット情報およびそれらに基づいて出力するトリガー判定情報を L1 buffer から読み出し、BCID や L1ID とともに SiTCP 通信にて送信する。同様に TTCFOB も L1A を受けた際の BCID、L1ID、ECRID などの TTC 情報を SiTCP 通信で送信する。12 台の New SL と 1 台の TTCFOB が出力するこれらのデータは 10 GbE のネットワークスイッチによって 1 本の線にまとめられ、SROD のソフトウェアが走る後段の PC に送信される。SROD PC ではソフトウェアを用いて受信したトリガーデータと TTC 情報を L1ID ごとにまとめて加工し、S-LINK Card という PCIe カードを用いて、S-LINK 通信でその加工データを ROS に送る。SROD PC は New SL クレートあたり 1 台、すなわち計 6 台用意されており、ROS は RobinNP Card という PCIe カードで SROD

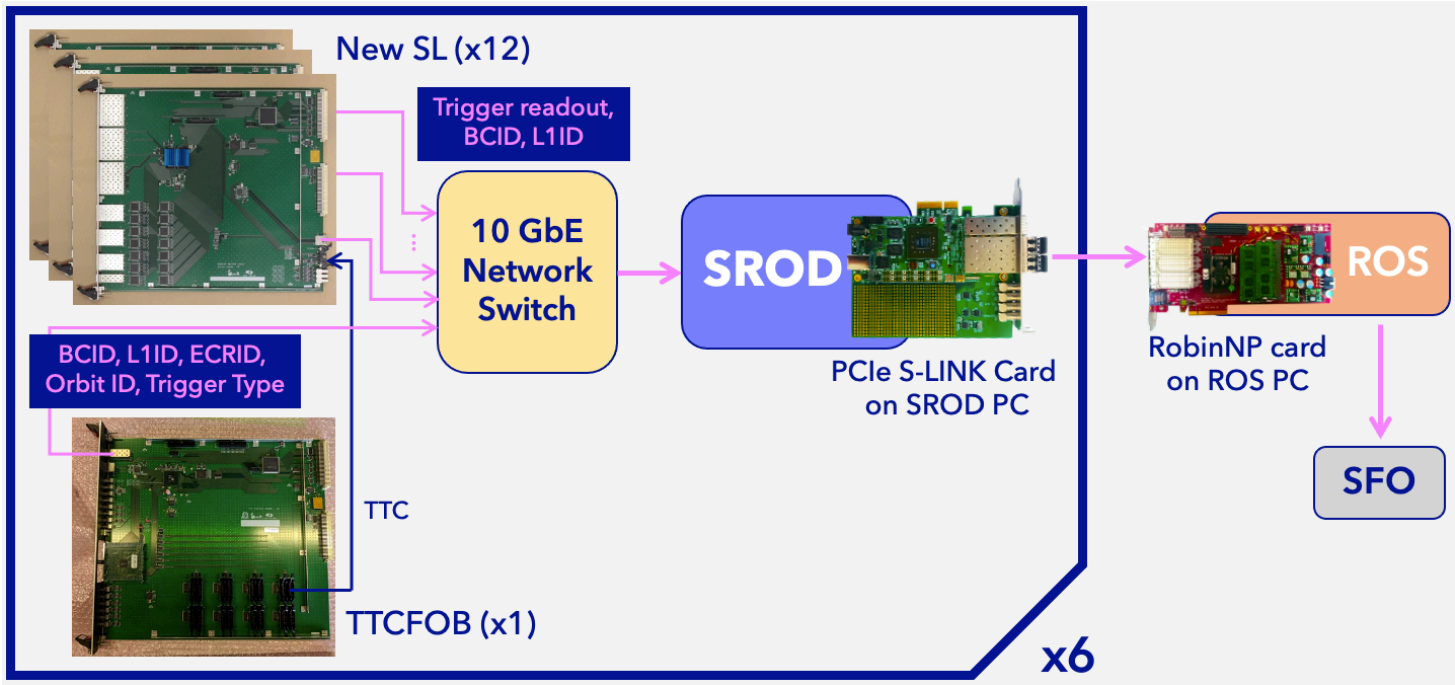


図 6.1　Run 3 における Level-1 ミューオンエンドキャップトリガーのデータ読み出しシステムの全体像。1 台の SROD は 12 台の New SL から送られてくるトリガーデータと 1 台の TTCFOB から送られてくる TTC 情報を L1ID ごとにまとめ、イベントビルディングを行う。全 6 台ある SROD のイベントデータは 1 台の ROS PC によってまとめられ、HLT の選別条件を満たしたイベントは SFO に送られ、記録される。

PC 6 台分のデータを受信する。これ以降のデータフローは図 3.17 にしたがい、読み出されたデータは SFO に記録される。

L1A は CTP によって 100 kHz の頻度で出力されるので、SROD PC は 100 kHz の頻度で 12 台の New SL と 1 台の TTCFOB からデータを受信する。まずこれらのデータをソフトウェアで用いるには、OS のカーネルレベルのバッファからそれらのデータを取得する必要がある。また各 New SL からデータを受信するタイミングは必ずしも同じではないため、SROD ソフトウェアでそれらのデータを揃えるようなバッファを用意しなければならない。さらに SROD は各 New SL と TTCFOB から送られてくるデータをイベントごとにまとめなければならないため、各データから L1ID を読み取る必要がある。最終的には、同じ L1ID を持つデータをまとめることで 1 イベント分の情報を ROS が受信できる形に加工し (イベントビルディング)、S-LINK Card を用いてそのイベントデータを送信しなければならない。すなわち、SROD ソフトウェアではこれらのすべての処理を合わせて、100 kHz の入力レートに耐えうる読み出しを実現することが要求される。

6.1.2　SROD ソフトウェアの構造

6.1.1 節で述べたように、SROD ソフトウェアはデータの受信、確認、加工、および送信のすべての処理を 100 kHz の入力レートに耐えうる速度で行う必要がある。これを実現するためにマルチプロセス方式を採用し、複数のアプリケーションがデータ処理を並列的に行うように SROD ソフトウェアを設計した。本小節では図 6.2 に沿って読み出しデータの処理手順およびマルチプロセスの制御について具体的に説明する。

ソケットバッファ

10 GbE ネットワークスイッチから 1 本のケーブルで送られてくる New SL 12 台分のトリガーデータと TTCFOB の TTC 情報はまず SROD PC 上の TCP ソケットバッファに入る。TCP ソケットバッファは SROD ソフトウェア上のものではなく、OS のカーネルレベルのものである。そのため、その大きさもカーネルパラメータによって決まっている。実際に物理データの取得に用いる SROD PC のソケットバッファの大きさは最大 4 MB に設定されている。

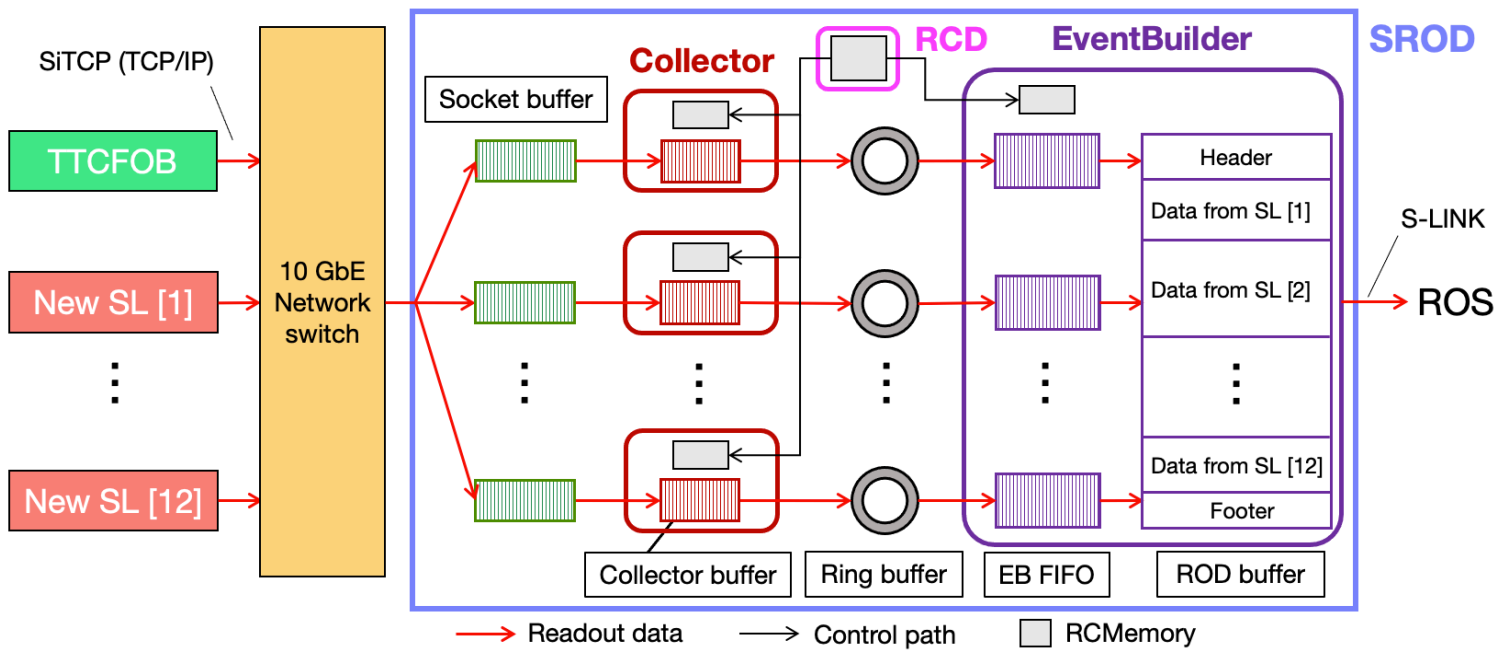


図 6.2　SROD ソフトウェアの構造。10 GbE ネットワークスイッチから送られてくるデータはまずソケットバッファに詰められる。各 Collector アプリケーションはソケットバッファに入っているデータを自身の Collector buffer に移動させ、そのデータを整えた後にリングバッファに送る。EventBuilder アプリケーションは各リングバッファからデータを取得し、対応する EB FIFO にデータを詰める。EB FIFO では L1ID や BCID などのデータが一部確認され、同じ L1ID のデータが ROD buffer に詰められる。最終的に ROD buffer 内の 1 イベント分のデータは S-LINK Card を用いて ROS へと送られる。Collector と EventBuilder を合わせた 14 個のアプリケーションが別プロセスとして並列に走る。これらのアプリケーションの動作は、RCD がそれぞれの RCMemory に RC Command を配布することで制御される。

Collector

Collector はカーネルレベルのソケットバッファからデータを取得し、ソフトウェアでの処理を可能にするためのアプリケーションである。12 台の New SL と 1 台の TTCFOB のそれぞれのデータを処理する計 13 個の Collector アプリケーションが別プロセスとして並列に走る。それぞれの Collector の処理手順は以下の通りである：

1. データ送信元の IP アドレスとポート番号を参照することでソケット通信を確立する。
2. 該当する New SL または TTCFOB のデータがソケットバッファに来るのを待つ。
3. ソケットバッファにデータが来たらそのデータを 32 ビット単位で読み出し、自身が持つ Collector buffer に詰める。
4. New SL と TTCFOB のデータ読み出しの実装の関係上、Collector buffer 内のデータは 16 ビットごとに 8 ビット単位で反転している[*1]。これを正しい順序に直す。
5. Collector buffer 内のデータをすべて後段のリングバッファに送る。
6. 2. に戻り、次のデータの処理に移る。

Collector アプリケーションの特徴は、イベントの境界を知らずにデータを扱うという点である。データの中身は特に見ず、ソケットバッファに詰まっているすべてのデータを一気に取得し、Collector buffer にて正しい順序に直し、一気にリングバッファに送るという役割を担っている。なお Collector buffer の大きさは現在 12.8 kB に設定しているが、これはソフトウェアレベルのパラメータなので自由に調整できる。

[*1] 例えば 32 ビットの 0x1234abcd というデータは 0x34→0x12→0xcd→0xab という順番で送られてくるため、Collector buffer には 0x3412cdab という形で詰め込まれる。これを元の 0x1234abcd に直さねばならない。

リングバッファ

リングバッファは SROD ソフトウェアによって Collector アプリケーションの数だけ用意されるバッファである。12 台の New SL と 1 台の TTCFOB からのデータは、主に TCP/IP 通信の影響により必ずしも同期して来るわけではない。すなわちすべての New SL と TTCFOB のデータが揃うまでの間、すでに来たデータを保持することができる、十分に大きいバッファが必要である。リングバッファがその役割を果たす。またリングバッファは Collector と後段の EventBuilder アプリケーションという 2 つのプロセス間の橋渡しを行わなければならないため、共有メモリとして用意されている。リングバッファの大きさは現在 1 MB に設定しており、これも SROD ソフトウェアによって自由に調整できる。

EventBuilder

EventBuilder は 12 台の New SL と 1 台の TTCFOB から来るデータの確認と加工、さらにはそのデータを ROS へ送信する役割を担う、SROD ソフトウェアの中心となるアプリケーションである。EventBuilder は次のような手順でデータ処理を行う[*2]：

1. ROD buffer に入っている、直前に処理された 1 イベント分のデータをクリアする。(Clear ROD buffer)
2. 自身が持つバッファである EventBuilder FIFO (EB FIFO) が空なら、対応するリングバッファから EB FIFO にデータを移動させる。該当するリングバッファにデータがまだないならば、それらが揃うのを待つ。データが揃ったらそれらを EB FIFO に移動させる。このときデータの中身は見ず、リングバッファに入っているすべてのデータを EB FIFO に移動させる。(Ring buffer to EB FIFO)
3. 12 台の New SL および 1 台の TTCFOB に対応する EB FIFO に入っているデータを順番に見ていき、1 イベント分のデータを探す。EB FIFO に入っている最後のデータまで見終わったら、対応するリングバッファからさらにデータを読み出す。(Find data in EB FIFO)
4. 各 EB FIFO で 1 イベント分のデータが見つかったら、そのデータから L1ID と BCID を読み取る。正常なデータならばすべての EB FIFO から読み取った L1ID と BCID は一致するはずなので、そうでない場合は ERS によってエラーメッセージを発行する。(Check event ID)
5. 各 EB FIFO で見つけた 1 イベント分のデータを ROS に送るためのフォーマットに加工し、ROD buffer に詰める。(Build event data)
6. SROD PC に挿入されている PCIe S-LINK Card を用いて、ROD buffer にあるデータを ROS PC に挿入されている RobinNP Card に送信する。(Send data to ROS)
7. 1. に戻り、次のイベントの処理を開始する。

このように EventBuilder アプリケーションは 1 イベント単位でデータの確認、加工、送信を繰り返し行う。EB FIFO と ROD buffer の大きさはそれぞれ 1 MB と 100 kB に設定しており、これらもソフトウェアレベルで自由に調整できる。

Run Control Memory (RCMemory) を用いたマルチプロセスの制御

上記の Collector アプリケーションおよび EventBuilder アプリケーションは別プロセスとして SROD PC 上で走る。これらのアプリケーションを系統的に制御するためには、第 5 章でも述べたように、RC State を用いた state transition を組み込む必要がある。ここではその方法について説明する。

まず、SROD ソフトウェア全体の RC State を司る Run Control Driver (RCD) は後述の SROD Segment を制御する Run Control Application である。よって 5.1.2 節でも説明したように、RCD は Partition の RootController から RC Command を受信する。次に、RCD は Collector や EventBuilder アプリケーションのそれぞれが持つ

[*2] 6.6 節にてこれらの処理を参照しやすくするために、括弧内にそれぞれの処理の名称を記す。

Run Control Memory (RCMemory) という共有メモリにこの RC Command を書き込む。最後に各アプリケーションは自身が持つ RC Memory に書き込まれた RC Command を読み込み、それに従って必要な動作を行う。このようにして RCD が RC Memory を用いて RC Command を各アプリケーションに配布することで、系統的な制御が行えるように設計した。

SROD ソフトウェア全体におけるイベントデータの処理時間

本研究では SROD ソフトウェアの性能評価を行い、イベントデータの処理時間などについて議論する。ここではそれにあたって、SROD ソフトウェア全体の処理時間について改めて俯瞰的に説明する。上記の 13 個の Collector と 1 個の EventBuilder を合わせた 14 個のアプリケーションは別プロセスとして並列に走る。各アプリケーションの処理 (上述した Collector の 1.–6. および EventBuilder の 1.–7.) は逐次的に行われる。よってプロセス間のデータ移動時間を無視すれば、並列して走るこれらの 14 個のアプリケーションのうち、1 イベント分のデータあたりの処理時間が最も長いアプリケーションの処理時間が SROD ソフトウェア全体の処理時間になる。特に EventBuilder は、1 つのプロセスで逐次的に 13 個のデータを処理するため、各 Collector より処理時間が長い。つまり SROD ソフトウェア全体の処理時間は、EventBuilder の処理時間とほぼ等しい。すなわち SROD ソフトウェアの性能は EventBuilder の性能によって決まる。EventBuilder の性能評価については 6.6 節で詳細に議論する。

6.2 OKS とソフトウェアアプリケーションの相互開発による読み出しパスの開通

SROD ソフトウェアはこれまで、CERN のテストベンチにある 1 台の PC を用いて開発されてきた [40]。Run 3 では New SL クレート 1 つにつき 1 台、すなわち全 6 台の PC 上で SROD ソフトウェアを走らせ、Level-1 ミューオンエンドキャップトリガーのすべての読み出しデータを処理しなければならない。本研究では SROD の OKS とソフトウェアアプリケーションを相互開発することで、Partition を用いて実際に物理データの収集に用いる 6 台の PC 上で SROD ソフトウェアを走らせ、100 kHz の頻度でデータを読み出すことに成功した。本節ではこれを実現するために行なった開発について述べる。

6.2.1 OKS の開発による SROD の全台制御の実現

第 5 章で述べたように、Partition を用いてオンラインソフトウェアアプリケーションを制御するには、それと整合するような形で OKS を開発する必要がある。SROD ソフトウェアの場合は全 6 台の SROD PC 上で Collector や EventBuilder アプリケーションを、TDAQ ソフトウェアの RC State と同期した形で走らせるための OKS が必要となる。本小節ではこれらの OKS の開発について説明する。

アプリケーションオブジェクトの定義

TDAQ ソフトウェアを用いて Collector や EventBuilder などの SROD のソフトウェアのアプリケーションを制御するためには、5.3 節で説明した New SL などの新規エレクトロニクスの場合と同様に、対応するオブジェクトを OKS データベースに定義しなければならない。ここではそれらの定義についてまとめる。

SROD のソフトウェアアプリケーションの定義の例として、図 6.4 に EventBuilder アプリケーションを定義している部分を示す。ここではまず Binary class の SRODEventBuilder というオブジェクトを定義することで、オンラインソフトウェアパッケージに用意した EventBuilder アプリケーションを OKS から参照できるようにしている。次にその Binary class のオブジェクトを使用する SROD-EventBuilder-A1 という ResourceApplication を定義することで、実際に該当する SROD PC 上で EventBuilder アプリケーションが走るようにした。SLCollector や TTCCollector についても同様に Binary および ResourceApplication を定義し、全 6 台の SROD PC 上で走るソ

```
<obj class="Binary" id="SRODEventBuilder">
 <attr name="BinaryName" type="string">"appEventBuilder"</attr>
 <attr name="Description" type="string">"Application that formats event data."</attr>
 <attr name="HelpURL" type="string">"http://"</attr>
 <attr name="DefaultParameters" type="string">""</attr>
 <rel name="Needs" num="0"></rel>
 <rel name="BelongsTo">"SW_Repository" "l1mue_sw_repository"</rel>
 <rel name="Uses" num="0"></rel>
 <rel name="ProcessEnvironment" num="0"></rel>
 <rel name="ExactImplementations" num="0"></rel>
</obj>
⋮
<obj class="ResourceApplication" id="SROD-EventBuilder-A1">
 <attr name="Lifetime" type="enum">"Configure_Unconfigure"</attr>
 <attr name="AllowSpontaneousExit" type="bool">1</attr>
 <attr name="Parameters" type="string">"SROD-EventBuilder-A1"</attr>
 <attr name="RestartParameters" type="string">"SROD-EventBuilder-A1"</attr>
 <attr name="Logging" type="bool">1</attr>
 <attr name="InputDevice" type="string">""</attr>
 <attr name="InitTimeout" type="u32">0</attr>
 <attr name="ExitTimeout" type="u32">5</attr>
 <attr name="StartIn" type="string">""</attr>
 <attr name="RestartableDuringRun" type="bool">1</attr>
 <attr name="IfExitsUnexpectedly" type="enum">"Error"</attr>
 <attr name="IfFailsToStart" type="enum">"Error"</attr>
 <rel name="InitializationDependsFrom" num="0"></rel>
 <rel name="ShutdownDependsFrom" num="0"></rel>
 <rel name="Program">"Binary" "SRODEventBuilder"</rel>
 <rel name="ExplicitTag">"" ""</rel>
 <rel name="Uses" num="0"></rel>
 <rel name="ProcessEnvironment" num="1">
   "Variable" "TGCSROD_ENV_SIDE_A1"
 </rel>
 <rel name="RunsOn">"Computer" "pc-tgc-rod-lvl1-01.cern.ch"</rel>
</obj>
```

in muons/sw/l1mue_sw_repository.data.xml

図 6.3　OKS における SROD ソフトウェアのアプリケーションの定義。ここでは EventBuilder アプリケーションを定義している部分を示している。Binary class の SRODEventBuilder オブジェクトを定義し、それを使用する SROD-EventBuilder-A1 という ResourceApplication を定義することで、SROD A1 の EventBuilder を動作させられるようになる。

フトウェアアプリケーションに対応する、計 84 個の ResourceApplication を用意した。この開発によって、6 台の SROD PC のそれぞれで 12 個の SLCollector、1 個の TTCCollector、そして 1 個の EventBuilder アプリケーションが動作する環境を構築した。

SROD Segment の開発によるソフトウェアアプリケーションの統合制御

次に、SROD PC 上で走る 14 個のアプリケーションの統合制御を行うために、新たに定義した ResourceApplication を取りまとめる SROD 用の Segment を用意しなければならない。そこで図 5.13 の青枠内にすでに示したように、TGC-A および TGC-C Segment のそれぞれに 3 つの SROD ソフトウェア用の Segment (TGCSROD-SideXX_Segment, XX=A1, A2, A3, C1, C2, C3) を導入した[*3]。各 SROD Segment は 1 台の SROD PC 上で走るソフトウェアアプリケーションをまとめており、Partition の RC State を各アプリケーションに伝播させる役割を持つ。それぞれの Segment が司るアプリケーションが走る SROD PC、および担当する New SL クレートの ID と TGC の 1/12 セクターの対応表を表 6.1 に示す。

SROD Segment の構造は 6 つとも同じなので、ここでは図 6.4 に示す SROD Segment A1 を例に挙げてその内部構造について説明する。TGCSROD_RCD-A01-A04 およびその配下にある L1MuESRODModule-A01-A04 は、図 5.11 に示した L1MuESRODModule パッケージのソフトウェアアプリケーションを走らせるための Run Control Application で、SROD ソフトウェアの各アプリケーションの RC State の管理などの全体的な制御を担う。ROL-TGC-SL-00-670011 とは、SROD のために新たに用意された ROS PC 上の RobinNP Card と通信を行うために必要な Resource である。12 個ある SROD-SLModule-AXX-XXX は New SL のデータをソケットバッファから取得する Collector アプリケーション (SLCollector) を表すために新たに定義した、ResourceApplication

[*3] 図 5.13 にて disable されている方の SROD ソフトウェア用の Segment は試験用に導入したもので、実際の運転では使用しない。

表 6.1　SROD Segment の名称と担当する New SL の対応表。

SROD Segment の名称	SROD PC の名称	担当する New SL クレートの ID	担当する TGC の 1/12 セクター
A1	pc-tgc-rod-lvl1-01	CSL04	A01, A02, A03, A04
A2	pc-tgc-rod-lvl1-02	CSL05	A05, A06, A07, A08
A3	pc-tgc-rod-lvl1-03	CSL06	A09, A10, A11, A12
C1	pc-tgc-rod-lvl1-04	CSL01	C01, C02, C03, C04
C2	pc-tgc-rod-lvl1-05	CSL02	C05, C06, C07, C08
C3	pc-tgc-rod-lvl1-06	CSL03	C09, C10, C11, C12

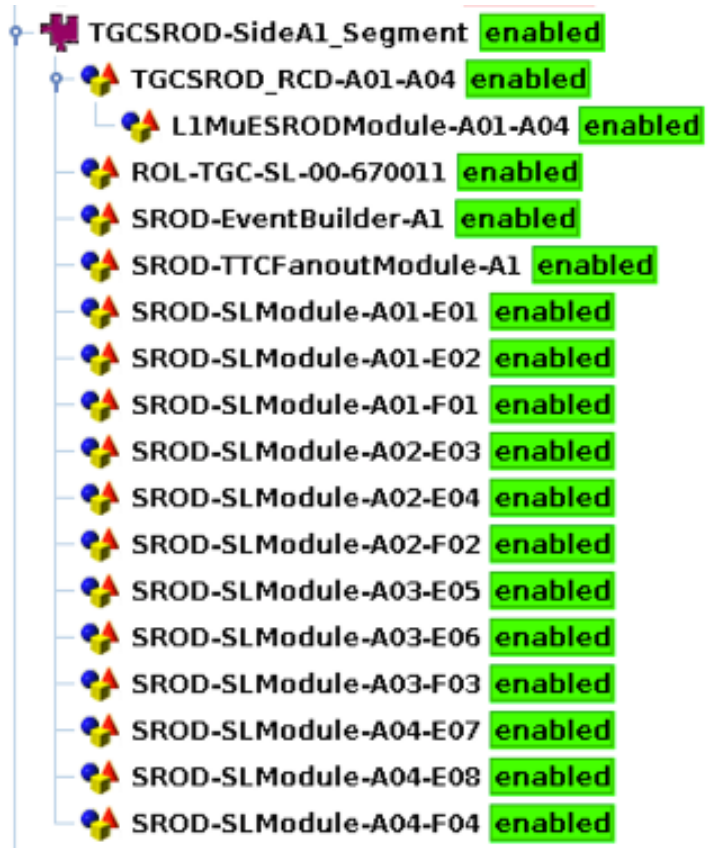


図 6.4　SROD Segment の構造。SROD ソフトウェア全体の制御を担う Run Control Application、ROS との通信を可能にする Resource、および SROD PC 上で走る Collector と EventBuilder に対応する ResourceApplication から構成される。

という class のオブジェクトである。同様に、SROD-TTCFanoutModule-A1 は TTCFOB から来るデータを取得する Collector アプリケーション (TTCCollector) に対応する。最後に、SROD-EventBuilder-A1 は EventBuilder アプリケーションを表すために定義した ResourceApplication である。6 台の SROD PC のそれぞれで Collector と EventBuilder アプリケーションを走らせるために、このようにして OKS 上にそのアプリケーションの数だけ ResourceApplication オブジェクトを用意した。また SROD のソフトウェアアプリケーションではここで説明した Segment の構造を参照して制御が行われるように実装した。詳しくは 6.2.2 節に後述する。

コンフィギュレーションパラメータの管理

5.1.1 節で説明したように、頻繁に変更するコンフィギュレーションパラメータは OKS データベースで管理することが望ましい。SROD ソフトウェアの場合はコミッショニングの観点から、メインアプリケーションである Collector と EventBuilder の動作を容易に変更できるようにすることで効率的に試験を進めることができる。そのため本研究ではこれらのアプリケーションのコンフィギュレーションパラメータを OKS で管理するように実装し

```
<obj class="SRODCollectorMode" id="SRODCollectorMode">
 <attr name="SLCollectorDumpMode" type="bool">0</attr>
 <attr name="TTCCollectorDumpMode" type="bool">0</attr>
 <attr name="SLCollectorDataCheck" type="bool">0</attr>
 <attr name="TTCCollectorDataCheck" type="bool">0</attr>
 <attr name="SLCollectorTimeMeasurement" type="bool">1</attr>
 <attr name="TTCCollectorTimeMeasurement" type="bool">1</attr>
</obj>

<obj class="SRODEventBuilderMode" id="SRODEventBuilderMode">
 <attr name="TruncateSLData" type="bool">0</attr>
 <attr name="WriteDummySLData" type="bool">1</attr>
 <attr name="WriteDataToFile" type="bool">0</attr>
 <attr name="SendDataToRos" type="bool">1</attr>
 <attr name="TimeMeasurement" type="bool">1</antr>
</obj>
```

in muons/segments/TGC/SROD/SROD_configuration.data.xml

図 6.5　SROD アプリケーションの動作モードの定義。これは data ファイルにてオブジェクトを定義している部分である。例えば SRODEventBuilderMode のパラメータである “SendDataToRos” は、ROS へのデータ送信を行うか否かを決める。勿論、物理データを取得する際は SendDataToRos を 1 に設定し、ROS へデータ送信を行わなければならない。他のパラメータについての説明は省略する。

た。ここでは Collector と EventBuilder のそれぞれについて、これまで開発してきた class とオブジェクトの概要をまとめる。

まず Collector アプリケーションは、New SL あるいは TTCFOB とソケット通信を確立してそれらのデータを取得するために、データの送信元の IP アドレスとポート番号を知っている必要がある。これらの情報を OKS 上で管理できるように、表 6.2 に示すような SRODCollectorModule という class を新たに定義した。SRODCollectorModule はそのパラメータとしてデータ送信元の IP アドレスとポート番号を持っている。対応する ResourceApplication と同じ名前を持つ 72 個の SLCollector と 6 個の TTCCollector 用の SRODCollectorModule オブジェクトを data ファイル内で定義することで、これらの情報をソフトウェアアプリケーションから取得できるように実装した。

表 6.2　SRODCollectorModule class の定義。SLCollector と TTCCollector はともに同じ種類のパラメータを持つ。

パラメータの名称	概要
IPaddress	データ送信元 (New SL または TTCFOB) の IP アドレス。
PortNumber	データ送信元のポート番号。

次に、コミッショニングで Collector と EventBuilder の動作の切り替えを効率的に行うために用意した class とオブジェクトについて説明する。Collector の動作モードを操作するために SRODCollectorMode、そして EventBuilder の動作モードを操作するために SRODEventBuilderMode という class および同名のオブジェクトを用意した。図 6.5 にそれらのオブジェクトを定義している部分を示す。それらのパラメータはすべて boolean 型として用意してあり、各操作をソフトウェアで行うか否かを決めるパラメータになっている。6.2.2 節に後述するように、オンラインソフトウェアではこれらのパラメータを CONFIGURE の段階で取得する。

本小節で説明した SROD ソフトウェアの OKS の開発についてまとめる。まずは SROD のソフトウェアアプリケーションに対応する 84 個のアプリケーションオブジェクトを定義した。またこれらのアプリケーションオブジェクトをまとめる SROD Segment を開発することで、各アプリケーションを TDAQ ソフトウェアの RC State と同期した形で走らせられるようにした。これらの 2 つの開発によって、Level-1 ミューオンエンドキャップトリガーのすべてのデータを読み出すために必要なソフトウェアアプリケーションが全 6 台の SROD PC 上で動作する環境を構築した。さらに、Collector と EventBuilder の動作を OKS から変更できるようにすることで、6.2.2 節で説明す

るソフトウェアアプリケーションの開発をより効率的に行えるようにした。

6.2.2 ソフトウェアアプリケーションの開発による 100 kHz でのデータ読み出しの実現

まず Collector や EventBuilder アプリケーションを含む SROD ソフトウェアを 6 台の SROD PC 上で走らせるためには、6.2.1 節で説明した OKS と整合した形で、SROD のソフトウェアアプリケーションにおけるコンフィギュレーションを実装する必要がある。これに加え、物理ランにおける L1A の出力頻度 (L1A レート) である 100 kHz でデータ読み出しを実現するために EventBuilder の性能改善を行なった。本小節ではこれらのソフトウェアアプリケーションの開発について説明する。

開発した OKS と整合したコンフィギュレーションの実装

前述のように本研究では 6 台の SROD PC 上で SROD ソフトウェアを走らせるために、OKS にて 6 つの SROD Segment を用意し、コンフィギュレーションパラメータを管理できるようにした。そこでソフトウェアアプリケーション側には、用意したパラメータを取得してコンフィギュレーションを行うようにした。また SROD Segment 内の ResourceApplication の enable/disable をソフトウェア側から参照するように実装することで、様々な用途での試運転、特に読み出しシステムの一部のみを用いた特定機能の試験が容易にできるような工夫を行なった。

まずは Collector アプリケーションのコンフィギュレーションについて述べる。Collector アプリケーションは RC State が CONFIGURE のときに起動するように OKS で設定してある。起動直後は自身の RCMemory を作成し、RCD の RCMemory からそのときの RC State と RC Command を取得することで Run Control FSM に沿って動作する。CONFIGURE では図 6.5 に示した Collector の動作モードを取得したのち、それぞれの Collector がデータを送る先であるリングバッファを作成するようにした。CONNECT では表 6.2 に示した IP アドレスとポート番号を OKS から取得し、対応する New SL または TTCFOB とのソケット通信を確立し、読み出しデータを取得する準備を行うように実装した。そして RC State が RUNNING になったら、6.1.2 節で説明した Collector の 2.–6. の動作が繰り返し実行されるようにした。これらの動作はすべて 5.4 節で説明した、New SL クレートのエレクトロニクスの state transition とも整合性が取れており、この実装によって New SL と TTCFOB から正しい手順でデータを取得できるようになった。

次に EventBuilder アプリケーションのコンフィギュレーションについて説明する。EventBuilder アプリケーションも Collector と同様に、CONFIGURE で起動した直後に RCMemory を作成することで、Run Control FSM に沿った動作を始める。CONFIGURE ではまず、Collector の ResourceApplication の Segments & Resources における enable/disable 状況を把握する必要がある (図 6.4 参照)。これはなぜかというと、仮に New SL あるいは TTCFOB のいずれかが何らかの理由で使用できなくなった場合、それに対応する Collector アプリケーションは disable することになる。するとこの Collector に対応するリングバッファは作成されないため、EventBuilder 側ではそれを見ないようにしなければならない。この動作の切り替えを実現するために、EventBuilder ではまず Segments & Resources の SROD Segment 内で enable されている Resource をすべて見て、enable されている SLCollector と TTCCollector をリストして管理できるように実装した。EventBuilder の以降の動作ではこのリストに基づいて動作を行うようにしてある。リストを作成した後は、図 6.5 に示した EventBuilder の動作モードを取得する。次に、Collector が作成したリングバッファをすべて開いて、読み出しデータを受け取れるようにした。そして RC State が RUNNING になると、6.1.2 節で説明した EventBuilder の 1.–7. の動作が繰り返し実行されるようにした。特に Collector アプリケーションの enable/disable を参照したコンフィギュレーションの実装は、ソフトウェアアプリケーションと OKS の連動した制御を可能にし、コミッショニング時の人為的なミスを削減するという点で非常に有用である。

このようにして Collector と EventBuilder アプリケーションのオンラインソフトウェアに OKS との整合性が取れたコンフィギュレーションの処理を実装することで、USA15 に新たに設置された 6 台の SROD PC で SROD

```
-   void EventBuilder::createRODFormat(ReadOutUtility* p_ru)
+   void EventBuilder::createRODFormat(ReadOutUtility* p_ru) // modified to avoid using std::stringstream (20201004)
    {
-     std::stringstream ss;
-     //ss << (p_ru->eb_buf_info).sl_id; //SLID from OKS
-     ss << (p_ru->slFormat->getSLID()); //SLID from SL Data
-     m_RODFormat->registerSLData(p_ru->slFormat->getBody(), ss.str());
+     m_RODFormat->registerSLData(p_ru->slFormat->getBody(), p_ru->slFormat->getSLID());
    }
```

図 6.6　EventBuilder の性能改善のために変更した処理。赤色でハイライトしている部分の処理を緑色でハイライトしている部分の処理に変えた。以前は std::stringstream を用いて string 型として New SL から送られてくる SLID を保持していた。しかし std::stringstream を定義する処理で約 0.4 μs もの時間を要していることがわかったので、SLID を整数型として別の関数に渡すようにした。これを含め多数の変更を加えることで、SROD ソフトウェアを用いて 100 kHz でデータを読み出すことが可能になった。

ソフトウェアを走らせる環境を構築した。これらの開発によって、全 72 台の New SL と全 6 台の TTCFOB から読み出しデータを 6 台の SROD PC で受信し、そのデータに適切な処理を加えてから ROS および SFO にデータを送ることが可能になった。すなわち、新しい読み出しシステムである SROD の統合運転環境への実装が実現し、Run 3 で TDAQ のインフラストラクチャーを用いて実際に物理データを収集する環境で、Level-1 ミューオンエンドキャップトリガーのデータ読み出しパスが開通した。

EventBuilder を中心とした読み出しパスの性能改善による 100 kHz でのデータ読み出しの実現

SROD の OKS とソフトウェア開発によって確立したデータ読み出しパスの検証を行うために、L1A レートを物理データの収集を行う場合と同じ 100 kHz に設定して SROD ソフトウェアを走らせた。しかしながら初期の段階では、EventBuilder アプリケーションの処理速度が 100 kHz の入力レートに耐えられず、正常に動作しなかった。この状況を打開するべく、本研究では EventBuilder を中心とした読み出しパスの性能改善を行い、100 kHz でのデータ読み出しを実現した。ここではそれらの性能改善に大きく貢献したものを抜粋して説明する。

L1A レートが 100 kHz の場合、EventBuilder は平均して 10 μs 以内に 1 イベントの処理を完了しなければならない。しかしはじめは、EventBuilder の処理時間は 15 μs 程度になっていた。この処理時間はリングバッファにデータが来るまでの待ち時間も含むので、EventBuilder の処理が間に合っているのであれば 10 μs 程度になるはずである。この処理時間の内訳を詳しく調べたところ、New SL から来るデータを識別するために用いていた std::stringstream という型の変数をインスタンス化する処理で、1 回あたり約 0.4 μs もの時間を要していることが判明した。この処理は 1 台の New SL から来るデータに対して 1 イベントあたり 1 回行なっていたため、1 イベントあたり合計で 5 μs 程度の時間をこの処理に割いていた。図 6.6 に示すように、ここの処理では stringstream 型ではなく整数を用いて New SL のデータを識別するように実装し、EventBuilder の処理時間を大幅に削減することに成功した。

加えて、EventBuilder のふるまいを調べることで New SL や TTCFOB の FPGA ファームウェアの読み出しパスの改善も行なった。EventBuilder では New SL や TTCFOB から送られてくるイベントデータの厳密なチェックを行う。これらのデータに何かしらの不整合がある場合、ERS メッセージの発行などの例外処理が実行される。はじめは New SL や TTCFOB から送られてくるデータが正しくなかったため、例外処理がすべてのイベントデータに対して行われ、EventBuilder の処理時間が延びていた。その例外処理の原因を調査して New SL と TTCFOB の FPGA ファームウェアを修正することで、読み出しパスの性能改善を行なった。

この他にも遅延が生じる処理について詳細に調査し、特に時間を要する処理を 1 つずつ洗い出して修正を行うことで EventBuilder を中心とした読み出しパスの処理時間を短縮し、100 kHz の高レートでトリガーデータを読み出せるようにした。今回開発した EventBuilder の具体的な性能について 6.6 節で詳しく議論する。

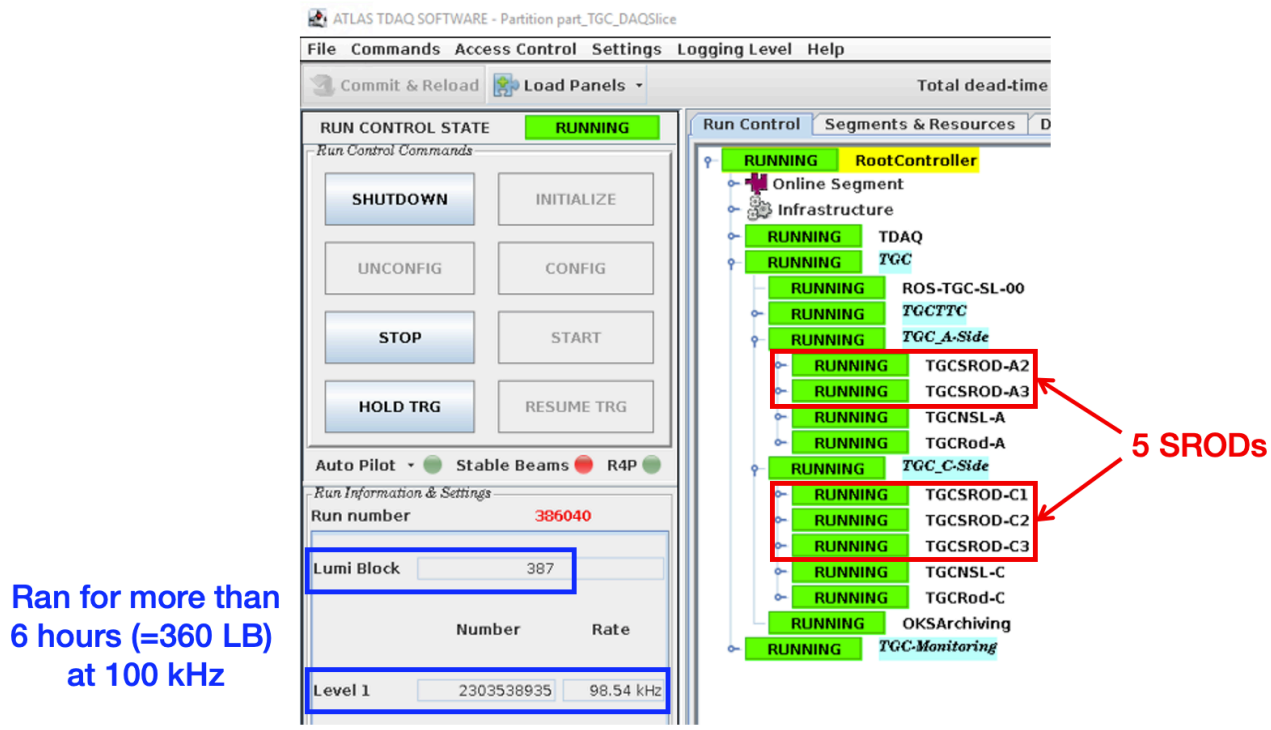


図 6.7　トリガーデータ読み出しパスの 100 kHz における長期安定性試験の際の Run Control Panel。1 LB は 1 分に相当する。5 台の SROD を用いた読み出しパスが 6 時間以上正常に動作することが確認できた。

100 kHz における長期安定性の検証

上記の EventBuilder を中心とした性能改善によって実現した読み出しパスの長期安定性を調べるために動作検証試験を行なった。この動作検証試験では DAQSlice Partition を用いて、全 6 台の SROD PC のうち SROD A1 を除く 5 台において SROD ソフトウェアを長時間走らせた[*4]。各 SROD PC は 12 台の New SL のそれぞれから 12 バイトおよび 1 台の TTCFOB から 20 バイトのイベントデータを受信するようにした。L1A レートは 100 kHz に設定し、読み出したデータの一部は SFO に記録されるようにした。なお ROS の XOFF は S-LINK Card で確認していなかったが、正常にデータが送信できていることは IS 経由で確認した。

試験中の Run Control Panel の写真を図 6.7 に示す。この試験では 5 台の SROD を用いた読み出しパスが 100 kHz という高い入力レートに対しても 6 時間以上正常に動作することが確認できた。これによって本研究で確立した読み出しパスの 100 kHz における長期安定性が確認できた。

6.3　Busy 信号の出力処理の実装による CTP とのハンドシェイクの実現

SROD ソフトウェアの処理速度が L1A レートに追い付かない場合、SROD ソフトウェアにて自動的に busy 信号を出力し、CTP における L1A の出力を一時的に止めることで、データのオーバーフローを防ぐ必要がある。SROD による busy 信号の出力は 4.2.4 節でも述べたように、SROD PC に挿入されている PCIe S-LINK Card を用いて行う。本研究では S-LINK Card を用いた busy 信号の出力処理を SROD ソフトウェアに実装することで CTP とのハンドシェイクを実現し、SROD ソフトウェアのデータ処理状況によって L1A レートを制御できるようにした。本節ではこの SROD ソフトウェアへの busy 信号の出力処理の実装について述べる。

[*4] 諸事情により SROD A1 が使用できない状態にあったので、それを除く 5 台で試験を行なった。

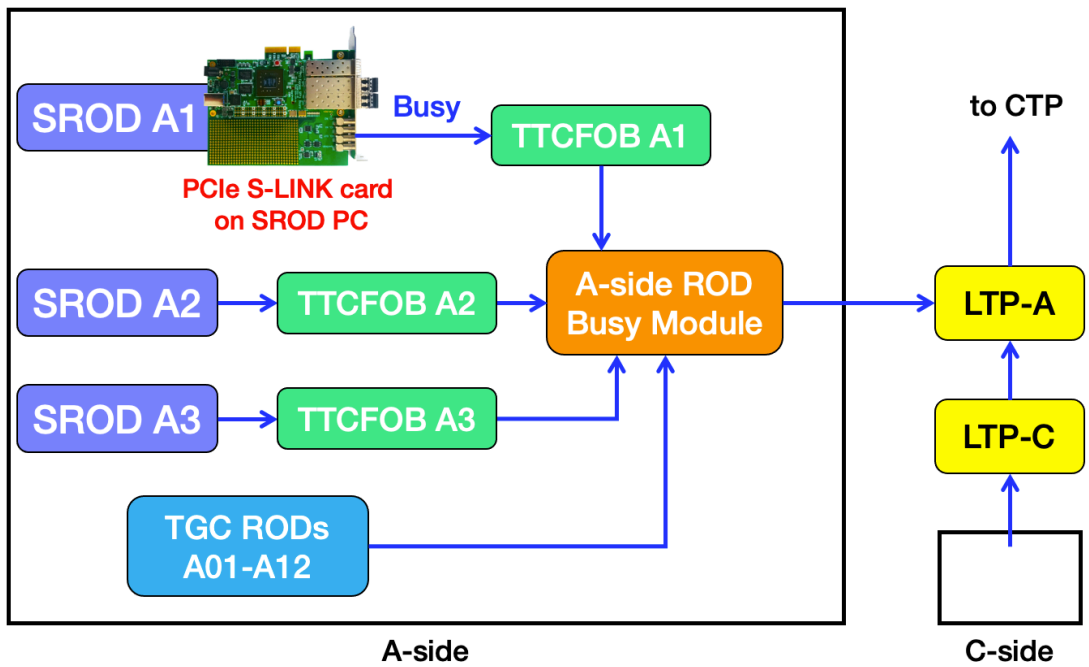


図 6.8　SROD に関係する busy 線の配線。SROD はその PC に挿入されている PCIe S-LINK Card を用いて busy 信号を出力する。この busy 信号は TTCFOB、ROD Busy Module、LTP を経由して、最終的に CTP に伝播される。CTP はこの busy 信号を検知することで、L1A の出力を一時的に止める。

6.3.1　Busy 線の配線

SROD の busy 信号を正しく伝播して L1A の出力を一時的に止めることができるようにするために、SROD PC から LTP までの busy 線を新たに配線した。図 6.8 にその配線図を示す。まず SROD ソフトウェアは SROD PC に挿入されている PCIe S-LINK Card を用いて、オープンドレイン方式で busy 信号を TTCFOB へと出力する。TTCFOB では SROD から来た busy と TTCFOB が持つ読み出し FIFO などの busy との OR が取られる。次に、TTCFOB は A-side と C-side のそれぞれに 1 つずつある ROD Busy Module に NIM 規格で busy 信号を送信する。ROD Busy Module は片サイドに 12 台ある TGC ROD と 3 台ある TTCFOB の busy 信号の OR を取り、LTP にその busy 信号を送信する。A-side の LTP は C-side の LTP からも busy 信号を受信し、最終的に Level-1 ミューオンエンドキャップトリガーおよび TGC 全体の busy 信号を CTP に送る。このように busy 線を正しく繋ぐことで、SROD が出力する busy 信号が CTP まで送られるようにした。

上記で説明した busy 線が正しく配線できているかを確認するために、簡単な試験を行なった。TGC の Standalone Partition である DAQSlice Partition を用いて、L1A レートを 100 kHz に指定し、全 6 台の SROD PC を用いて SROD ソフトウェアを走らせた。Run が始まって A-side の LTP から L1A が出力され始める[*5]前に EventBuilder アプリケーションに busy 信号を止める処理を実装し、必ず L1A が出力されるようにした。その状況から、各 SROD PC において S-LINK Card から busy 信号を出力するコマンドを手動で順番に送り、L1A の出力が止まるかを Run Control Panel から確認した。そのときの Run Control Panel を図 6.9 に示す。この試験によって 6 台の SROD PC 上の S-LINK Card から出力された busy 信号が LTP にて正しく検知されたことが確認できた。

6.3.2　OKS における busy 線の定義

Partition を走らせた際に L1A の出力を正しく行うためには、OKS データベースにて busy 線を定義し、Segments & Resources にて busy 線の enable/disable を制御できるようにしなければならない。Busy 線の定義を実際の配線

[*5] DAQSlice Partition には CTP は含まれず、A-side の LTP が L1A を出力する。

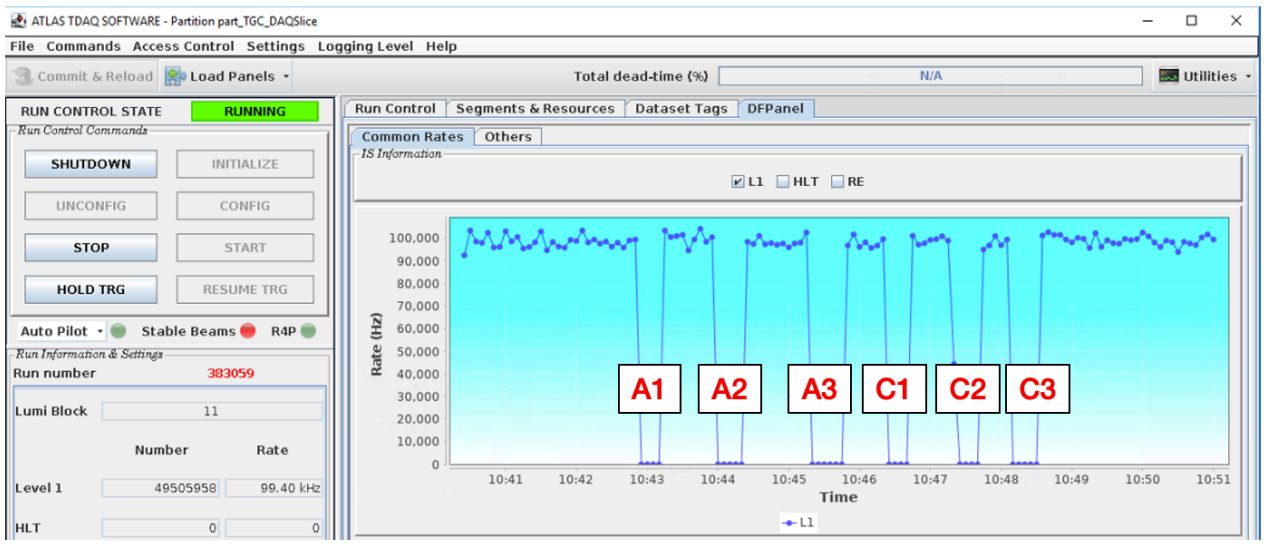


図 6.9　SROD に関係する busy 線の配線の確認試験の際の Run Control Panel。これは L1A レートの時間変化を表している。6 台の SROD PC の S-LINK Card から手動で busy 信号を出力したところ、確かに L1A レートが 0 になり、busy 線が正しく配線されていることが確認できた。

と合わせて正確に行わないと、意図しない busy 信号によって L1A の出力を、つまりは物理データの収集を止めることになってしまう。そこで本研究では Segments & Resources を用いて SROD の busy 信号の enable/disable を制御できるようにし、ソフトウェアアプリケーションが OKS と連動した形で busy 線のマスクをできるような実装を行なった。

Level-1 ミューオンエンドキャップトリガーおよび TGC システムにおいては、図 6.8 に示したように、ROD Busy Module が ROD と TTCFOB の busy 信号をまとめる。すなわち SROD が出力する busy 信号を制御するためには、ROD Busy Module において新たに配線した TTCFOB の busy 線の入力を enable/disable できるようにすれば良い。それを可能にするために、図 6.10 に示すように、TTCFOB の busy 信号線の入力を表す 3 つの Resource を ROD Busy Module の ResourceSet の配下に新たに置いた。この Resource を enable/disable することで、TTCFOB の busy 信号、すなわち SROD の busy 信号を enable/disable することができるように実装した。この実装により、ソフトウェアアプリケーションが OKS と連動して全 6 台の SROD PC の S-LINK Card から出力される busy 信号を制御をできるようになり、大規模システムがより安定して動くようになった。

6.3.3　SROD ソフトウェアを用いた busy 信号の出力処理の実装

SROD におけるデータ処理速度が L1A レートに追い付かない場合は、SROD ソフトウェアにて自動的に busy 信号を出力し、L1A レートを抑制する必要がある。本研究では EventBuilder アプリケーションにその機能を実装した。EventBuilder におけるデータの処理が遅れると、リングバッファからデータを取得する速度が遅れる。しかしリングバッファにデータを詰める Collector は別プロセスとして走っているため、EventBuilder のデータ処理速度が L1A レートに間に合っていない場合は、リングバッファに詰まっている未処理のデータが増えていくような設計になっている。そこでリングバッファの使用状況に応じて busy 信号を出力する処理を EventBuilder アプリケーションに実装した。

EventBuilder のデータ処理の手順は 6.2 節で述べた通りである。EventBuilder は 12 台の New SL および 1 台の TTCFOB のそれぞれのデータを保持する計 13 個のリングバッファからデータを読み出す。この読み出し処理を行うときに、次のような手順でリングバッファの使用状況をモニターし、状況に応じて busy 信号を出力する処理を組み込んだ：

1. リングバッファが busy であるか否かを表す busy フラグをすべて下げる。
2. 13 個ある Collector に対応するリングバッファの使用状況を順番に見ていき、使用状況が 50% 以上ならばそ

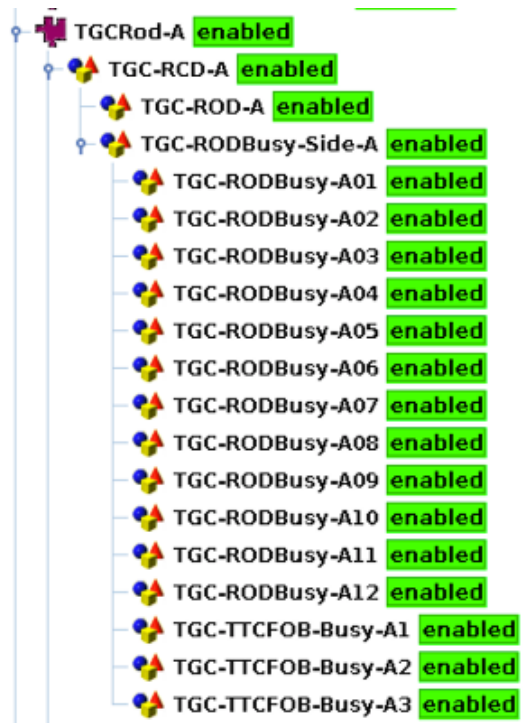


図 6.10　Segments & Resources における ROD Busy Module の busy 信号入力の定義。TGC-RODBusy-Side-A という ResourceSet の配下には ROD Busy Module の busy 信号線の入力を表す Resource が配置されている。TTCFOB の busy 信号、すなわち SROD の busy 信号を enable/disable できるようにするために TGC-TTCFOB-Busy-AX (X=1,2,3) を新たに定義した。C-side についても同様に実装した。

のリングバッファの busy フラグを立てる。

3. 13 個あるリングバッファの busy フラグのうち 1 つでも立っていれば、S-LINK Card を用いて busy 信号を出力し、その原因となったリングバッファを ERS の Warning メッセージとして表示する。ただしすでに busy 信号を出力している場合は改めて busy 信号および Warning メッセージの出力を行わないようにした。逆にリングバッファの busy フラグがすべて下がっていて、以前の処理によって busy 信号が出力されている場合は、ここで busy 信号の出力を止めるようにした。
4. 1. から 3. の処理をイベントデータが来るたびに、つまりは L1A レートと同じ頻度で繰り返し行う。

なお 2. における 50% という数字は現時点では暫定的に設定したもので、今後この値は OKS データベースから設定できるように実装し、busy 信号の伝播時間などに応じて最適化を行う。

6.3.4　SROD ソフトウェアを用いた busy 信号の出力処理の動作検証

上記のようにして SROD ソフトウェアに実装した busy 信号の出力処理が正しく動作するかを確認するために、EventBuilder の処理が L1A レートに間に合わないような設定で SROD ソフトウェアを走らせた。この試験では DAQSlice Partition を用いて、A-side の LTP にて L1A が 100 kHz の頻度でランダムに出力されるように OKS で設定してある。この L1A を受信した A05 から A08 セクターの 12 台の New SL と TTCFOB A2 は SROD A2 に読み出しデータを送信する。それぞれの New SL からは固定長のデータを送るようにしてあり、SROD A2 から ROS へのデータ送信量が 1 イベントあたり 11232 ビットになるようにした[*6]。なお ROS が S-LINK Card に送信する XOFF は無視するような状態で試験している。

図 6.11 に上記の動作試験中の Run Control Panel を示す。ERS メッセージの欄には EventBuilder アプリケーションが発行した、リングバッファの busy 信号が出力された旨を示すメッセージが表示されている。また L1A レー

[*6] 具体的には、12 台の New SL のそれぞれから MUCTPI に送信するミューオン候補 7 つ分のダミー情報を固定で送信している。よって SROD から ROS へのデータ送信量は、480 bits + 32 bits × 7 × 4 BC × 12 = 11232 bits となる。計算方法については 6.4.1 節を参照していただきたい。

図 6.11　SROD ソフトウェアの busy 信号の出力処理の動作検証試験における Run Control Panel。赤枠で囲っている ERS メッセージの欄にリングバッファの使用状況によって busy 信号を出力した旨の Warning メッセージが表示されている。青枠で囲っている部分は L1A レートの時間変化を表している。L1A は 100 kHz の頻度で出力されるように設定しているが、busy 信号の自動出力により実際の出力頻度は 80 kHz で頭打ちになっている。

トを 100 kHz に設定しているにもかかわらず、実際の L1A レートは約 80 kHz で頭打ちになっており、100 kHz を下回っている。このことから、SROD ソフトウェアがそのリングバッファの使用状況に基づいて自動的に busy 信号を出力していることが間接的に確認できた。今回の試験では SROD A2 が処理するデータ量は 6.4.2 節に後述する Run 3 における平均データ量より遥かに多くなるように設定しており、EventBuilder の処理が間に合わないような状況になっているので、これは想定通りのふるまいである。すなわちこの試験で busy 信号の出力処理が正しく行われ、LTP とのハンドシェイクが正しく実現されていることが確認できた。この試験では DAQSlice Partition を使用しているので CTP は含まれていないが、LTP が受信した busy 信号はそのまま CTP に伝播されるので、ATLAS Partition を使えば直ちに同じふるまいが再現される。

上記の動作試験では S-LINK Card にて ROS の XOFF 信号を確認しない設定で試験を行なっているが、実際の物理ランではデータ送信時に ROS の XOFF を確認し、データを確実に送信できるようにする。今回実装したリングバッファの使用状況による busy 信号の出力は、ROS の XOFF を見るような設定でも可能である。しかし実際に XOFF が来た場合は、EventBuilder が行う ROS にデータを送信する処理において、データが送信できるようになるまで待機する。EventBuilder は逐次的に処理を行うので、この待機時間の影響でリングバッファをモニターして busy 信号を出力する処理が遅れる可能性がある。このとき Collector は変わらずしてリングバッファにデータを詰めるので、場合によってはリングバッファにてデータのオーバーフローが起こる。今後はこのような状況に対処できるような busy 信号の出力処理を新たに追加する予定である。

6.4　トリガー読み出しデータのフォーマットの検討

SROD は New SL と TTCFOB から 100 kHz の頻度で送られてくる読み出しデータをまとめてイベントビルディングを行い、形成したイベントデータを ROS に送信する。ROS に送信されたデータは HLT によって選別されたの

図 6.12　TTCFOB から SROD に送られるデータのフォーマット。

ち、オフラインで解析を行うために CERN のコンピューティングセンターである Tier-0 に半永久的に保存される。すなわち SROD が ROS に送信するデータはオフライン解析に必要な情報をすべて含まねばならない。そこで本研究では以下の要件を満たすように、SROD の入出力データのフォーマットを検討した：

- SROD でのイベントビルディングに必要な情報をすべて含むような入力データフォーマット
- オフラインでのトリガー性能評価およびデバッグのために必要な情報をすべて含む、ROS が受信できるような出力データフォーマット

本節ではこれらの新しい SROD の入出力データフォーマットについて説明する。また 6.6 節で性能評価を行うにあたって、これらのデータフォーマットに基づいて入出力データの大きさを議論する。

6.4.1　SROD の入力データ

1 台の SROD PC は 1 台の TTCFOB および 12 台の New SL から読み出しデータを受信する。TTCFOB からは L1ID や BCID などの TTC 信号の情報が送られてくる。New SL からは各検出器エレクトロニクスから受信したヒット情報 (トリガー論理回路の入力情報) とトリガー判定情報 (トリガー論理回路の出力情報) が送られてくる。本研究ではこれらの SROD の入力データのフォーマットを検討し、それらの大きさを見積もったので、この小節ではそれらについてまとめる。

TTC Fanout Board から SROD に送られるデータ

TTCFOB は L1A を受信すると、対応する BC における TTC 情報を SiTCP 通信を用いて SROD に送信する。図 6.12 にそのデータのフォーマットを示す。これらの TTC 情報については 3.2.7 節ですでに述べたので、その内容についての説明は省略する。TTCFOB から受信したこれらの L1ID や BCID を、EventBuilder アプリケーションにて ROS に送信するイベントデータに付与するように実装した。

TTCFOB は上記の 160 ビットの TTC 情報を 100 kHz の頻度で SROD に送信する。すなわち TTCFOB から SROD へのデータの送信量は、

$$160\ \text{bits} \times 100\ \text{kHz} = 16.0\ \text{Mbps} \tag{6.1}$$

となる。

New SL から SROD に送られるデータ

New SL は TGC BW、TGC EI/FI、NSW、RPC BIS78、および Tile Calorimeter から送られてくるヒット (トラック) 情報を基にトリガー判定を行い、MUCTPI にそのトリガー判定情報を送る。これらのトラック情報およびトリガー判定情報はすべて一時的に L1 buffer に保存され、L1A が来た際に L1ID や BCID とともに読み出され、ゼロサプレスロジックによって圧縮されてから SROD へと送られる。ここではトラック情報やトリガー判定情報の具体的なフォーマットには深入りせず、その概観についてのみ述べる[*7]。

New SL から SROD に送られるデータは各検出器エレクトロニクスから New SL に送られてくるミューオント

[*7] より詳しいデータフォーマットについては付録 E.1 で補足する。

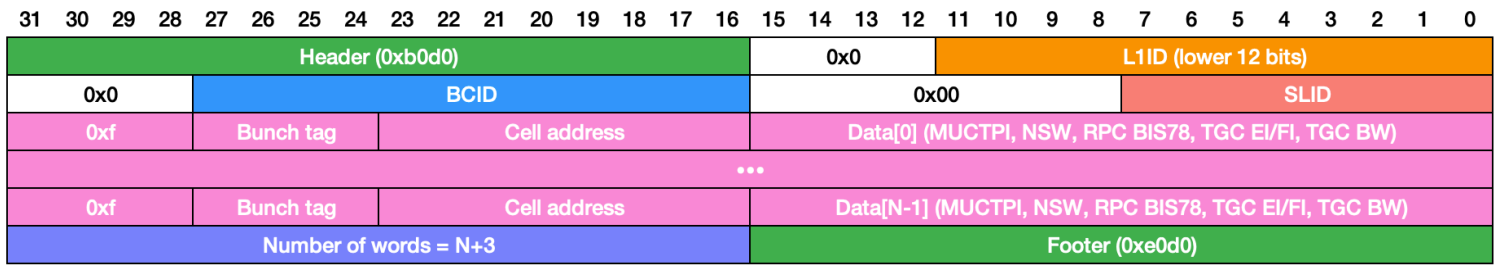


図 6.13　New SL から SROD に送られるデータのフォーマット。New SL は L1A を受けると、L1ID の下位 12 ビット、BCID、1 クレート内の間で New SL を識別するために用いる 8 ビットの SLID、そして送信した 32 ビットデータの数を含む少なくとも 96 ビットの情報を必ず SROD に送る。図の桃色の部分は MUCTPI に送信するトリガー判定情報や各検出器エレクトロニクスから受信したトラック情報を表している。この部分はミューオン候補の数によって毎回異なるので、読み出しデータの大きさは可変である。

ラックの候補数によるので、その大きさは可変である。そのデータフォーマットの概要を図 6.13 に示す。New SL は L1A を受けると少なくとも 96 ビットのヘッダーとフッター情報を SROD に送る。この 96 ビットには L1ID、BCID、SLID、そして送信したデータ数が含まれる。これらの情報は EventBuilder アプリケーションにてイベントビルディングを行うために使用するために用意したが、ROS に送信するデータには組み込まれない。実際に ROS に送信するデータに組み込まれる MUCTPI に送信したトリガー判定情報や各検出器のトラック情報に関してはやや複雑なので、以下で議論する。

New SL は L1A 信号を受信するとその L1A に対応する current BC に加え、その前後の previous/next BC と 2 つ後の after next BC の計 4 BC 分のトラック情報を L1 buffer から読み出し、SiTCP 通信にて SROD PC に送信する。しかし CTP が L1A を出力した要因は必ずしも Level-1 ミューオンエンドキャップトリガーであるとは限らない。仮にその L1A の要因が Level-1 ミューオンエンドキャップトリガーであったとしても、トラック候補の数は最大になるとは限らない。すなわち、L1A が来た際に読み出されるトラック情報のほとんどは空、つまりゼロデータである可能性が高い。よって New SL の FPGA ではゼロサプレスロジックを用いてデータを圧縮し、SROD に送信するデータ量を削減している。New SL のゼロサプレスロジックでは全ビットが 0 であるトラック情報はすべて無視され、ビットが立っているトラック情報のみ SROD に送られるようになっている。データ圧縮後のトラック情報が何に対応するかを特定するために、図 6.13 に示したように、各トラック情報には 0xf、Bunch tag[*8]、およびセルアドレスから構成される 16 ビットの “ゼロサプレスヘッダー” が付与される。Run 3 の物理ランにおけるゼロサプレス圧縮率はおよそ 10^{-3} 程度であると見積もられている [41]。このような形で New SL のトリガー論理回路のすべての入出力データを効率的な手法で読み出すことで、トリガー性能評価やデバッグなどのオフライン解析を行うのに十分な情報を安定して SROD PC に送信できるようにした。

では Run 3 の物理ランで New SL から SROD に送られる平均データ量を計算する。まずは表 6.3 に New SL が送信する 1 BC あたりの最大のデータ量を示す。これらの値は、各検出器エレクトロニクスから送られてくるトラック情報および MUCTPI に送るトリガー判定情報にゼロサプレスヘッダーを付加することを考えることで計算できる[*9]。最終的にエンドキャップおよびフォワード New SL はそれぞれ 1 BC あたり最大で 2432 ビットおよび 1792 ビットのトラック情報とトリガー判定情報を SROD に送る。マージンを持たせてゼロサプレス圧縮率を 10^{-2} と仮定すると、1 台の New SL が SROD に送る平均データ量は、

$$\begin{aligned} \text{エンドキャップ}: (96\ \text{bits} + 2432\ \text{bits} \times 4\ \text{BC} \times 10^{-2}) \times 100\ \text{kHz} = 19.3\ \text{Mbps} \\ \text{フォワード}: (96\ \text{bits} + 1792\ \text{bits} \times 4\ \text{BC} \times 10^{-2}) \times 100\ \text{kHz} = 16.8\ \text{Mbps} \end{aligned} \tag{6.2}$$

と計算される。

[*8] ここで言う Bunch tag とは BCID とは異なり、読み出す 4 BC 分のデータを識別するための ID である。Previous、current、next、after next のそれぞれに 0x0、0x4、0x8、0xc が割り当てられている。

[*9] 前述したように、詳細は付録 E.1 に記す。

表 6.3　New SL が SROD に送る 1 BC あたりの最大データ量。TGC BW、TGC EI/FI、Tile Calorimeter、および MUCTPI のゼロサプレス後のデータはミューオン候補 1 つあたり、32 ビットである。NSW と RPC BIS78 のゼロサプレス後のデータはミューオン候補 4 つあたり 192 ビットである。

データの種類	1 台の New SL あたりの ミューオン候補の最大数 (エンドキャップ / フォワード)	1 BC あたりの 最大データ量 [ビット] (エンドキャップ / フォワード)
MUCTPI	8	256
NSW	24	1152
RPC BIS78	4 / -	192 / -
TGC EI/FI	3 / -	96 / -
TGC BW	22 / 12	704 / 384
Tile Cal.	1 / -	32 / -
合計	-	2432 / 1792

最後に、Run 3 の物理ランにおいて 1 台の SROD PC が受信する平均データ量を計算する。1 台の SROD PC は 8 台のエンドキャップ New SL、4 台のフォワード New SL、および 1 台の TTCFOB からデータを受け取る。すなわち 1 台の SROD PC が受信する平均データ量は式 (6.1) と式 (6.2) より、

$$19.3\ \mathrm{Mbps} \times 8 + 16.8\ \mathrm{Mbps} \times 4 + 16.0\ \mathrm{Mbps} = 238\ \mathrm{Mbps} \tag{6.3}$$

と見積もることができる。ただしこの値は New SL におけるゼロサプレスロジックによる圧縮率を 10^{-2} と大きめに仮定した場合のものである。ゼロサプレス圧縮率を 10^{-3} として同じ計算をした場合、1 台の SROD PC が受信する平均データ量は 142 Mbps になる。

6.4.2　SROD の出力データ

SROD ソフトウェアは EventBuilder アプリケーションにて New SL や TTCFOB から来たデータを用いてイベントビルディングを行い、ROS が受信できるような形にイベントデータを整える。また EventBuilder は SROD PC に挿入されている S-LINK Card を用いてそのデータを ROS PC に挿入されている RobinNP Card に送る。6.1.2 節でも述べたように EventBuilder の性能は SROD ソフトウェア全体の性能に直接関係しており、特にこのデータ送信の処理に時間を要することがわかっている。このデータ送信時間はデータ送信量に依存するので、Run 3 の物理ランにおいて SROD から ROS にどれだけのデータを送るかを知ることは重要である。本小節では新たに検討した SROD の出力データのフォーマットについて説明したのち、送信データ量を計算する。

図 6.14 に本研究で検討した、EventBuilder アプリケーションが作る ROS への送信データのフォーマットを示す。ヘッダーとフッターは ATLAS が指定している共通の ROS フォーマットに沿っており、それらを合わせた計 480 ビットは固定で ROS に送られるようになっている。この 480 ビットには TTCFOB から送られてきた L1ID や BCID などの TTC 情報や、SROD の状態を含むようにしている。トラック情報の部分は New SL から送られてきたミューオン候補の数に依存する。MUCTPI、TGC EI/FI、TGC BW、Tile Calorimeter のトラック情報は 1 つのミューオン候補あたり 32 ビットのデータになる。また NSW と RPC BIS78 のトラック情報は 1 つのミューオン候補あたり 64 ビットの情報になる。このようにしてデータフォーマットを検討することで、オフラインで Level-1 ミューオンエンドキャップトリガーの性能評価やデバッグを行うために必要な情報が、デコードしやすい形で記録されるようにした。

では 6.6 節で性能評価を行うにあたって、SROD から ROS に出力するデータの大きさを計算する。ROS に送る最大データ量は、表 6.3 に示した 1 BC あたりのミューオン候補の最大数を用いて計算できる。その結果を表 6.4 に示す。エンドキャップ New SL およびフォワード New SL に関するトラック情報はそれぞれ 1 BC あたり最大

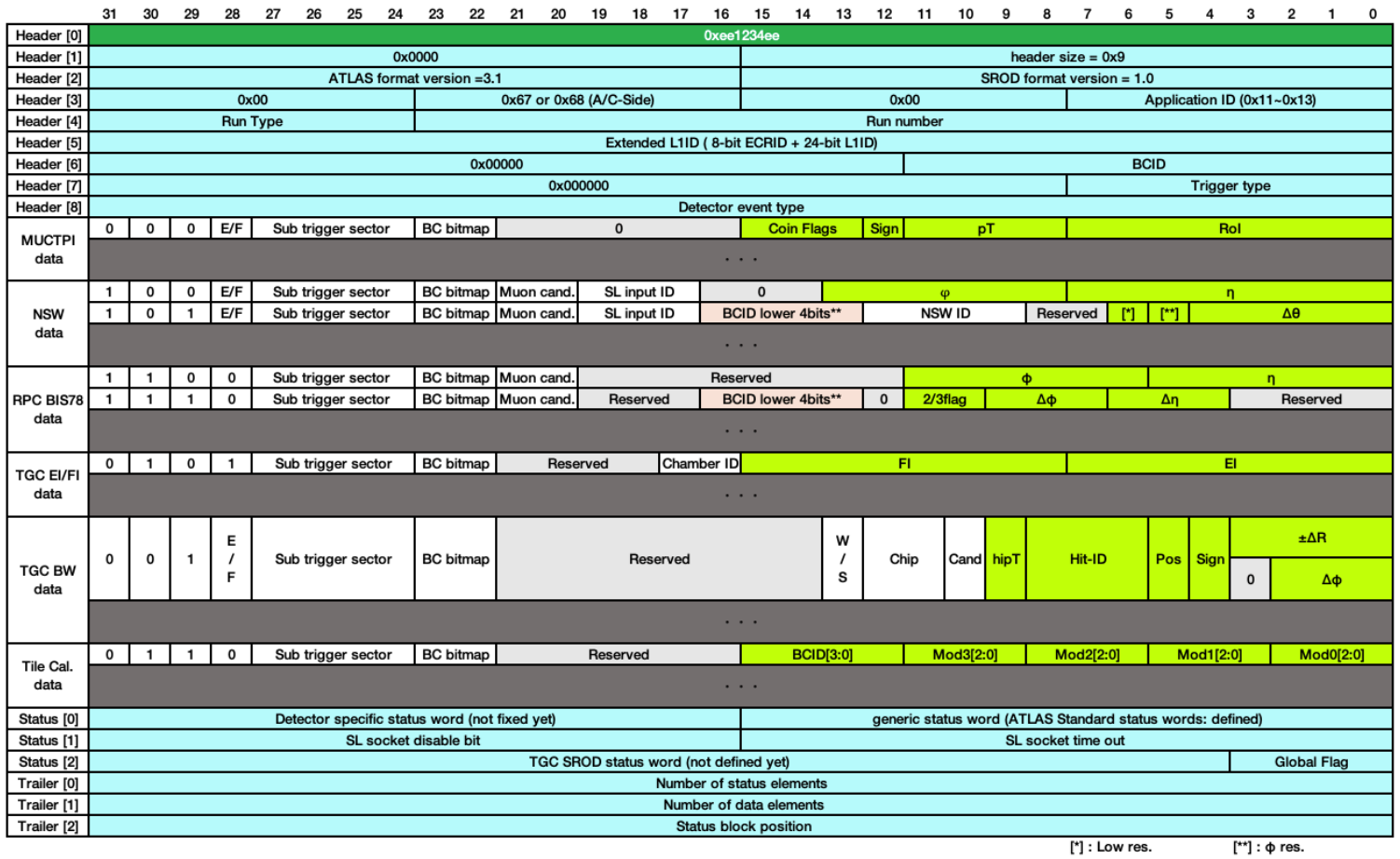


図 6.14　SROD から ROS に送るデータのフォーマット。288 ビットのヘッダーと 192 ビットのフッターを合わせた計 480 ビットは必ず送られる。TTCFOB から送られてくる TTC 情報はすべてこの 480 ビットに含まれる。New SL から送られてきたトラック情報は 1 つのミューオン候補あたり 32 ビットあるいは 64 ビットのデータに変形される。

で 2880 ビットおよび 2176 ビットと計算される。これよりゼロサプレス圧縮率を 10^{-2} と仮定した場合に 1 台の SROD PC から ROS に送られる平均データ量は、ヘッダーとフッターを合わせた 480 ビットも加味して、

$$\left\{480\ \text{bits} + (2880\ \text{bits} \times 8 + 2176\ \text{bits} \times 4) \times 4\ \text{BC} \times 10^{-2}\right\} \times 100\ \text{kHz} = 175\ \text{Mbps} \tag{6.4}$$

と計算される。なおゼロサプレス圧縮率を 10^{-3} と仮定して同じ計算を行うと、ROS へのデータ送信量は 60.7 Mbps になる。6.6 節ではこれらの値を参照して EventBuilder の性能について議論する。

表 6.4　SROD が ROS に送るトラック情報の最大データ量。

データの種類	1 つのミューオン候補あたりのデータ量 [ビット]	1 台の New SL あたりのミューオン候補の最大数 (エンドキャップ / フォワード)	1 BC あたりの最大データ量 [ビット] (エンドキャップ / フォワード)
MUCTPI	32	8	256
NSW	64	24	1536
RPC BIS78	64	4 / -	256 / -
TGC EI/FI	32	3 / -	96 / -
TGC BW	32	22 / 12	704 / 384
Tile Cal.	32	1 / -	32 / -
合計	-	-	2880 / 2176

図 6.14 に示したデータフォーマットへのデコード機能はすでに EventBuilder アプリケーションのソフトウェアに実装されている。その実装が正しいかを確認するために、現在もデータフォーマットのバリデーションを進めてい

る。具体的には New SL に複数パターンの入力データを埋め込み、SFO に記録された出力データが期待通りになっているか確認することで、入力データと出力データの間の整合性チェックを行なっている。このバリデーションによって、New SL のゼロサプレスロジックから SROD ソフトウェアの EventBuilder におけるデコードまでが正しく行われていることが確認できている。

6.5 L1A レートが低い場合における読み出しパスの安定性の検証

6.2 節で説明した SROD の OKS とソフトウェアの開発によって、実際に物理データを収集するときの L1A レートである 100 kHz でトリガーデータを読み出すことが可能になった。しかし ATLAS Partition などを用いた試運転では L1A レートが 100 kHz より低い場合もあるので、そのような場合での安定動作も保証せねばならない。そのために 100 kHz を下回る L1A レートで SROD ソフトウェアを走らせ、本研究で確立した読み出しパスの安定性を検証した。その過程で SROD ソフトウェアに由来しない読み出しパスの問題を発見したので、その問題の解決策を提案し、実装した。本節ではこれらの動作試験、および発見した問題とその解決策について議論する。

6.5.1 L1A レートが低い場合における動作検証試験について

この試験でも DAQSlice Partition を用いて、A-side の LTP から指定した頻度に応じてランダムに L1A を出力する設定で試験を行なった。L1A レートは 100 kHz より低い、100 Hz、400 Hz、1 kHz、4 kHz、8 kHz、10 kHz、40 kHz に指定し、それぞれで数回試験を行なった。A05 から A08 セクターの 12 台の New SL と TTCFOB A2 から読み出しデータを SROD A2 に送るようにした。ただしこの試験では New SL からトラック情報を模したデータは送らず、L1ID や BCID などの情報のみを送るようにした。読み出しパス全体の動作を確認したかったので、S-LINK Card では ROS の XOFF を正しく確認しながらデータ送信を行うようにし、SFO までデータが流れるようにした。

表 6.5 に動作試験の観測事項を示す。L1A レートが 100 Hz および 40 kHz のときは 100 kHz と同様に、正常に走った。しかしその間の 400 Hz から 10 kHz の場合は ROS にて “not deleted events” と呼ばれるエラーが生じた。Not deleted events とは、ROS の RobinNP Card がバッファに持つイベント情報を削除するように HLTSV から要求されたときに (clear request[*10])、その情報がバッファになかったため、削除できなかったイベントがあることを示すエラーである。しかしその削除要求の後に ROS にイベントデータが届いた場合、RobinNP Card のバッファにそのイベント情報が残る。このような遅延イベントが生じ続けると、遅れて届いたイベント情報によって RobinNP Card のバッファが占有され、ROS が新たなイベント情報を処理できない状況に陥り、XOFF 信号を出力する。今回の動作試験では、400 Hz で走らせた場合はわずか 15 分で ROS から XOFF 信号が S-LINK Card に出力され、それ以上データを送信することができなくなった。100 Hz の場合を除いて、L1A レートがより低いほど、not deleted events の頻度が高くなることが観測された。

Not deleted events が生じているということは、LTP によって L1A の出力が行われてから、SROD から ROS にその L1A に対応するトリガーデータが送られるまでの間の読み出しパスで、何かしらの原因によって遅延が生じていることを意味する。次の小節でその原因について説明する。

[*10] 図 3.17 参照。

表 6.5　L1A レートが低い場合における読み出しパスの動作試験の結果。それぞれの L1A レートで数回走らせて、観測された事項に再現性があるかを確認した。100 Hz の場合を除き、L1A レートが低いほど not deleted events がより多く生じた。

L1A レート	観測事項
100 Hz	正常に走った。
400 Hz	ROS にて not deleted events が頻繁に生じ、15 分程度で ROS から XOFF 信号が来た。
1 kHz	ROS にて not deleted events が生じたが、その頻度は 400 Hz の場合より少なかった。
4 kHz	ROS にて not deleted events が生じたが、その頻度は 1 kHz の場合より少なかった。
10 kHz	ROS にて not deleted events が稀に生じた。
40 kHz	正常に走った。

6.5.2　L1A レートが低い場合に生じる読み出しパスの遅延の原因

低い L1A 出力頻度において読み出しパスの遅延が生じる原因を解明するべく、試験した L1A レートのうち最も not deleted events の頻度が高かった 400 Hz で、SROD ソフトウェアが行う処理の時間測定を行なった。その結果、遅延の原因は SROD ソフトウェアではなく、TTCFOB と New SL から読み出しデータを送信する際に用いている SiTCP 通信の設定が関係していることがわかった[*11]。ここではその SiTCP 通信の設定について説明する。

SiTCP 通信を含む一般的な TCP/IP 通信は Nagle アルゴリズムというバッファリングアルゴリズムを採用することで、データ転送を効率的に行うようになっている。 Nagle アルゴリズムでは、TCP 送信バッファのデータ量が最大セグメントサイズ (Maximum Segment Size, MSS) に達した場合、または送信バッファへの書き込み操作がある時間間隔 (Nagle タイムアウト時間) の間ない場合、送信バッファ内のデータが送られるようになっている。SiTCP 通信では、デフォルトで Nagle アルゴリズムを使うようになっており、MSS は 1460 バイトに設定されている [32]。これらの設定はユーザーによって変更が可能である。また SiTCP の Nagle タイムアウト時間は 4 ms で [42]、こちらはユーザーが変更できない仕様になっている。

表 6.5 に示した観測事項も上記の SiTCP 通信のパラメータによって説明できる。L1A レートが 100 Hz の場合は 10 ms 間隔で SiTCP の送信バッファに書き込むため、4 ms で Nagle タイムアウトが起こり、New SL と TTCFOB のどちらにおいてもイベントごとのデータ送信が実現される。しかし L1A レートが 250 Hz より高い場合は Nagle タイムアウトが起こらないので、SiTCP 送信バッファのデータが MSS のデフォルト値である 1460 バイトに達するまでデータは送信されていなかった。New SL の 1 イベントあたりの読み出しデータは TTCFOB より小さい 12 バイトであったため、L1A レートが 400 Hz の場合は読み出しパスにおける遅延時間は最大で

$$\frac{1}{400\ \mathrm{Hz}} \times \frac{1460\ \mathrm{bytes}}{12\ \mathrm{bytes/event}} = 304\ \mathrm{ms} \tag{6.5}$$

程度であったと概算できる。L1A が出力されてから HLTSV が ROS へ clear request を送るまでの時間は明確には決まっていないが、今回の試験結果によってそのおおよその値も推測できる。表 6.5 によると L1A レートが 10 kHz の場合ではごくわずかに not deleted events が生じていて、40 kHz では生じていなかった。また 100 Hz では Nagle タイムアウト時間である 4 ms の遅延が起きていても not deleted events は生じていなかったことも考慮すると、読み出しパスに許される遅延時間は約 10 ms 程度であると予測できる。これは DAQSlice Partition を使った試験を参考に見積もった値だが、ATLAS Partition を用いた場合も同様である [43]。すなわち not deleted events を生じさせないためには、L1A レートに依らず、10 ms 以内に New SL と TTCFOB から読み出しデータを送るような工夫が必要である。

[*11] 原因調査の過程は付録 E.2 に記す。

6.5.3 L1A レートが低い場合に生じる読み出しパスの遅延時間の短縮方法の提案と実装

6.5.2 節で述べたように、遅延時間の原因は SiTCP 通信の Nagle アルゴリズムによるバッファリングにあることがわかった。そこで New SL と TTCFOB から読み出しデータを送るまでの時間を短縮するために、3 つの解決策を考えた：

1. SiTCP 通信の Nagle アルゴリズムをオフにしてバッファリングが一切行われないようにすることで、直ちに読み出しデータを送るようにする
2. SiTCP 通信の MSS をデフォルトの 1460 バイトから小さくすることで、バッファリングによる遅延時間を短くする
3. New SL および TTCFOB でイベントデータの間に “pad word” というダミーデータを埋め、TCP 送信バッファがのデータ量がすぐに MSS に達するようにすることで、バッファリングによる遅延時間を短くする

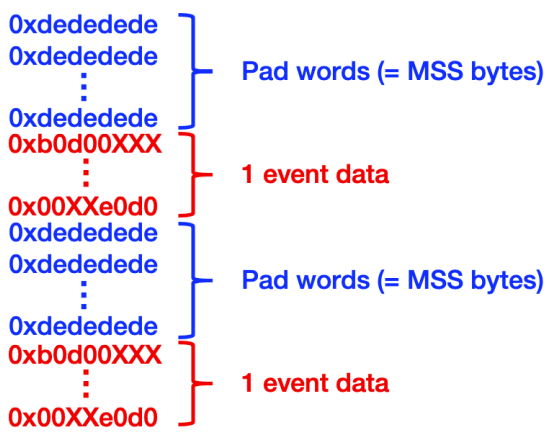
これらの解決策を試してみたところ、1. では SROD ソフトウェアの Collector にて壊れたデータが検知され、EventBuilder がエラー状態に陥り正常に動作しなかった。2. の場合は MSS を小さくしすぎると 1. と同じふるまいが観測され、MSS が適度に大きいと not deleted events がわずかに生じた。しかし MSS の最適化を行うにあたって許された範囲が 60–120 バイトと狭く、安定動作を保証することが困難であると判断した。これらの結果から、解決策の 1. と 2. は採用しないことにした[*12]。

上記の 3 つの解決策のうち、唯一正常にデータを読み出しつつ、遅延時間の短縮に成功したのは解決策 3. であった。この方法では MSS を縮小するのではなく、L1A レートが低い場合に限って New SL および TTCFOB から送信するデータを増やすことで、送信するまでの待ち時間を短縮するようにする。具体的には New SL と TTCFOB が送る 1 イベント分のデータの間に 32 ビットの “pad word” を詰めることで、SiTCP の送信バッファ内のデータ量がより早く MSS に達するようにする。New SL および TTCFOB にて単位時間内に MSS 分のデータが送られていなければ、図 6.15(a) に示すようにイベントデータの間に MSS 分の pad word を詰めるように実装する。このようにすることで、すでに安定動作が確認されている L1A レートが高い場合は pad word を送らずに済み、L1A レートが低い場合に限って pad word を詰めることができる。この提案に沿って New SL と TTCFOB の試験用の FPGA ファームウェアが現在も開発されている。

また SROD ソフトウェアには pad word を無視する処理を組み込まねばならない。SROD ソフトウェアにおけるデータの中身の確認は Collector では行わず、EventBuilder のみが行う。Pad word を無視するにはそのデータが pad word であることを識別する必要があるため、この処理は EventBuilder に実装した。図 6.15(b) に示すように、6.1.2 節で説明した EventBuilder の処理手順の 3. において、1 イベント分のデータの前に pad word があれば、それを飛ばす処理を付け加えた。

SROD ソフトウェアに実装した pad word を飛ばす処理が正しく動作し、実際に not deleted events が生じなくなるかを確認するために、簡単な試験を行なった。1 イベントの間に必ず指定したバイト数の pad word を送信する機能を持つ New SL と TTCFOB の試験用ファームウェアを用いた。1 台の New SL および 1 台の TTCFOB の MSS をデフォルト値の 1460 バイトに設定し、それぞれ 1 イベントあたり 12 バイトおよび 20 バイトの読み出しデータを SROD に送るようにした。これに合わせて、New SL から送信する pad word を 1448 バイト、TTCFOB から送信する pad word を 1440 バイトに設定し、1 イベントごとに SiTCP の送信バッファ内のデータ量が MSS に達するようにした。これらの設定で L1A レートを 400 Hz にして長時間走らせ、not deleted events が生じるかを確認した。その結果、図 6.16 に示すように、10 時間走らせても not deleted events は全く生じていないことが確認できた。すなわち pad word を導入することで L1A レートが低い場合に生じていた not deleted events をなくすこ

[*12] 解決策 1. と 2. の検証については付録 E.3 に詳細を記す。



(a) Pad word の送信方法

```
bool EventBuilder::readTTCDataFromFIFO(ReadOutUtility* p_ru)
{
  p_ru->fifo->setTmpOffset(p_ru->fifo->getReadOffset());

  //skip pad word (0xdededede)
  while( p_ru->fifo->getData( p_ru->fifo->getTmpOffset()) == 0xdededede ) {
    p_ru->fifo->incrementTmpOffset();
    if( p_ru->fifo->getTmpOffset() >= p_ru->fifo->getUsage()){
      try {
        if(!readDataFromRingBuffer(p_ru, p_ru->fifo->getTmpOffset())){
          SROD_WARNING_MSG("readDataFromRingBuffer failed during readTTCDataFromFIFO.");
          return false;
        }
      }
      catch( const IOException &e ){
        throw( ComponentBaseException( __FILE__ , __LINE__ ,  e.reason() ) );
      }
    }
  }
  p_ru->fifo->updateFinishedWord(p_ru->fifo->getTmpOffset()-p_ru->fifo->getReadOffset());
  p_ru->fifo->setReadOffset(p_ru->fifo->getTmpOffset());

  if( p_ru->fifo->getData( p_ru->fifo->getTmpOffset()) != TTCFormat::HEADER ){
    if(searchHeaderOfNextEvent(p_ru,TTCFormat::HEADER)){
      p_ru->error_code = m_srod_exception.ERROR_ONLY_CURRENT_HEADER_NOT_FOUND;
    }else{
      p_ru->error_code = m_srod_exception.ERROR_HEADER_NOT_FOUND;
    }
    m_error_flag->is_ttc_format_available = false;
    if (m_counter_daqRun % 100000 == 0) SROD_WARNING_MSG("Couldn't find the TTCData header.");
    return false;
  }
```

(b) EventBuilder に実装した pad word を飛ばす処理

図 6.15　Pad word の実装方法。図 6.15(a) は pad word の埋め込み方を表す。L1A レートが低い場合に限って、New SL と TTCFOB において 1 イベント分のデータの間に SiTCP の MSS のバイト数だけ pad word (0xdededede) を詰めるようにすることで、一定時間内にイベントデータが送信されることを保証する。図 6.15(b) は EventBuilder に実装した pad word を飛ばす処理。ここでは TTCFOB から来たデータにおいて pad word を飛ばす処理を見せている。赤枠で囲った部分が今回の実装で付け加えた部分である。TTCFOB のイベントデータのヘッダーを探す処理の前に、pad word を飛ばすようにしている。New SL から来るデータについても同様の処理を実装した。

とができることに加え、EventBuilder に実装した pad word を飛ばす処理が問題なく動作していることが確認できた。上記の試験結果に加え、L1A レートが 100 kHz など高い場合においても、pad word を飛ばす処理が入った状態で SROD ソフトウェアが正しく動作することも確認した。

次のステップは、一定時間内に MSS 分のデータが送信されなかった場合のみに pad word を埋め込む機能を New SL および TTCFOB のファームウェアに実装し、その動作検証を行うことである。すでにその機能のプロトタイプは試験用ファームウェアに実装され、動作検証試験が行われた。2021 年 1 月現在は、この機能のパラメータの調整や詳細なデバッグが進められている。

6.6　SROD の性能評価

6.2 節から 6.5 節までで説明した、本研究で確立したトリガーデータ読み出しパスおよび SROD ソフトウェアの堅牢性を検証するために、その性能評価を行なった。6.1.2 節でも述べたように、SROD ソフトウェアの性能は EventBuilder アプリケーションの性能によってほぼ決まるため、具体的には EventBuilder の性能を評価した。この性能評価は、SROD 単独の性能評価を行うための試験と、ROS と SFO を含む読み出しパス全体の性能評価を行うための試験の 2 通りの条件で行なった。なおこの 2 通りの条件を実現するために、具体的には SROD PC に挿入されている S-LINK Card にて ROS の状態 (XOFF 信号) をデータ送信時に確認しない場合とする場合でデータ収集を行なった。これらの 2 通りの条件で、次の 3 つの項目を調べた：

- EventBuilder の処理時間の分布と内訳
- EventBuilder の処理速度と ROS に送信するイベントデータの大きさとの関係
- EventBuilder の処理速度と L1A レートの関係

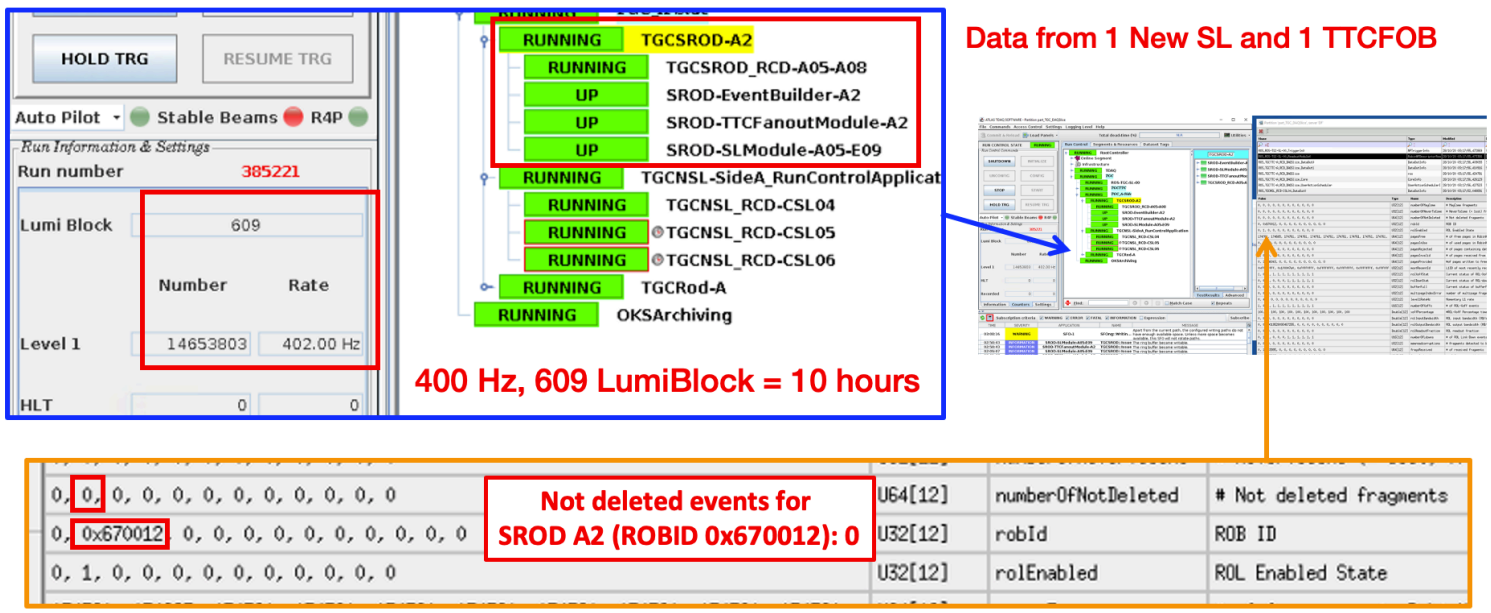


図 6.16　Pad word 処理の動作検証試験の結果。1 台の New SL と 1 台の TTCFOB から 1 イベントごとに MSS 分のデータが送られるように pad word を埋め込み、SROD ソフトウェアではそれらの pad word を無視するようにした。L1A レートを 400 Hz にして 10 時間走らせても、not deleted events が生じていないことが IS から確認できた。

本節では上記の評価項目について得られた結果を議論する。6.6.1 節で SROD 単独の性能評価試験 (ROS の XOFF を確認しない場合)、6.6.2 節で ROS と SFO を含む読み出しパス全体の性能評価試験 (ROS の XOFF を確認する場合) について述べる。

ここに 6.6.1 節と 6.6.2 節に共通する試験条件を記す。まず本節のすべての評価は、DAQSlice Partition を用いて、LTP-A からランダムに L1A が出力されるような設定で行なった。全 6 台の SROD は並列して走るので、性能評価の観点からは 1 台のみを走らせれば十分であることから、A05–A08 セクターを担当する 12 台の New SL と TTCFOB A2 が SROD A2 に読み出しデータを送信するようなセットアップを用いた。ROS から送られる XOFF の確認の有無にかかわらず、S-LINK Card を用いて RobinNP Card に読み出しデータを送信し、SFO に読み出しデータを記録した。使用したバッファの大きさは表 6.6 に示す通りである。使用した SROD ソフトウェアには 6.3 節で述べた busy 信号の出力処理、および 6.5 節で述べた pad word を飛ばす処理も含まれている。Busy 信号の出力はリングバッファの使用状況が 50% を超過した場合、すなわちリングバッファに 500kB 以上のデータが詰まっているときに行うようにした。New SL と TTCFOB の SiTCP の設定については、Nagle アルゴリズムはオンにして MSS は 1460 バイトに設定してある。

表 6.6　EventBuilder の性能評価試験における各バッファの大きさ。バッファの名称については図 6.2 を参照していただきたい。ソケットバッファを除くすべてのバッファの大きさは SROD ソフトウェアにて設定できる。ソケットバッファの大きさはカーネルパラメータである。なお SLCollector と TTCCollector のバッファの大きさは別々に調整できるが、今回は 12.8 kB に統一した。

バッファの名称	大きさ
ソケットバッファ	最大 4 MB
SLCollector buffer	12.8 kB
TTCCollector buffer	12.8 kB
リングバッファ	1 MB
EB FIFO	1 MB
ROD buffer	100 kB

6.6.1 SROD 単独の性能

まずは S-LINK Card にて ROS からの XOFF 信号を確認しない場合での性能、つまりは SROD 単独の性能について議論する。

EventBuilder の処理時間の分布と内訳

SROD ソフトウェアが行う処理の大部分を担う EventBuilder アプリケーションの性能によって SROD ソフトウェアそのものの性能が決まる。ソフトウェアの性能はそれを走らせる CPU の性能にも依るが、USA15 に実際の物理ランで用いる SROD PC がインストールされている現在、それらの実機を用いて EventBuilder が行う処理の所要時間を実測し、より理解を深めることが重要である。そこで本研究では EventBuilder が 1 イベントに対して行う処理を 6 つの処理ステップに分割して考え、それらの平均所要時間および処理時間の分布を実測した。これらの 6 つの処理ステップとは 6.1.2 節で説明した EventBuilder が行う処理の 1. から 6. に対応する。この処理時間はリングバッファにデータが来るまでの待機時間を含むように定義した。例えば 100 kHz でデータを読み出すときの平均処理時間は 10 μs となることが期待される。

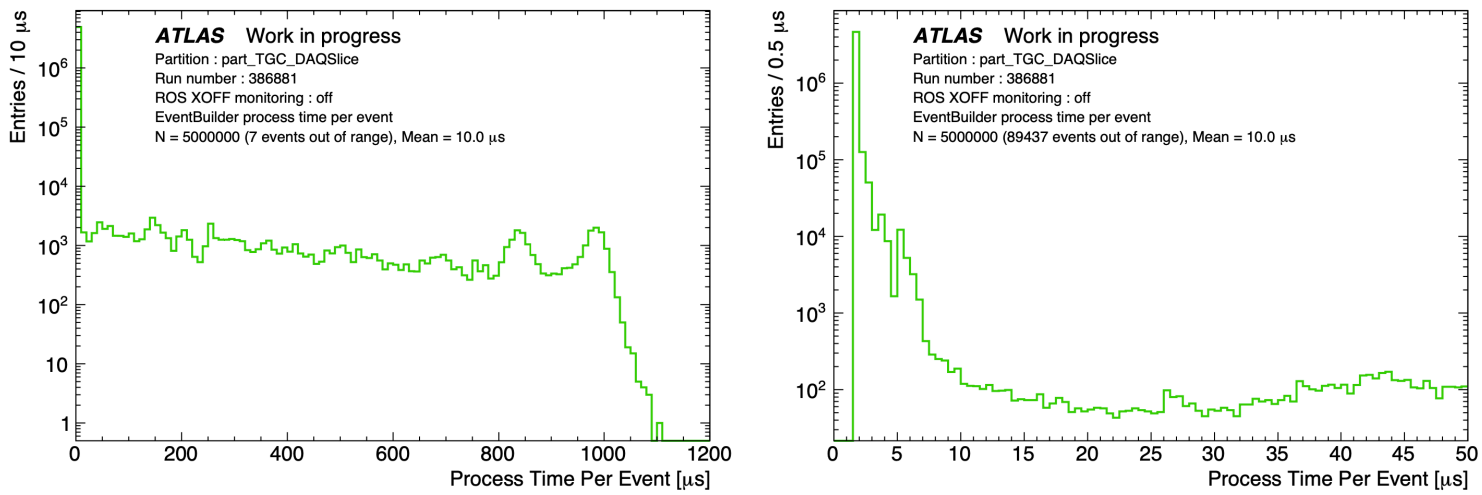
それぞれの処理時間の分布を調べるために、EventBuilder にて処理時間を測定するための処理を新たに導入した。各処理に要する時間をイベントごとに測定し、ns を単位として整数値で時間データを保持するようにした。これらの時間データをイベントデータの中にダミーデータとして埋め込み、ROS に送信した。そして SFO に記録されたデータをオフラインでデコードして解析することで、処理時間の分布を求めた。ただし DAQSlice Partition では高レートで SFO にすべてのイベントデータは記録できないので、今回の設定では ROS が受信したイベントの 1% がランダムで SFO に記録されるようにしていた。1 台の New SL (A05-E09) からは 96 ビットのヘッダーとフッターに加え、128 ビットのダミートリガー判定情報を送るように設定し、New SL から 6.4.1 節で説明したトリガーデータが送られていることを保証した。他の 11 台の New SL からは 96 ビットのヘッダーとフッターのみを送るようにしていた。L1A レートは 100 kHz に設定し、90 分間走らせて記録されたイベントのうち、最初の 500 万イベントを解析した。

まずは EventBuilder の全体処理時間の分布を図 6.17 に示す。これらの 500 万イベントの平均処理時間は 10.0 μs になっていたので、100 kHz の L1A レートと整合している。しかし 500 万イベントのうち約 10 万イベントで 10 μs 以上の処理時間を要しており、その処理時間分布のテールは約 1.2 ms まで続いていることがわかる。そのテールに影響している処理は図 6.18 に示す各処理ステップの所要時間の分布を見ることでわかる。Ring buffer to EB FIFO および Find data in EB FIFO の処理時間分布に同様のテールが見られる。これらの 2 つの処理は共通してリングバッファにデータが来るまでの待ち時間を含んでいるので、その待ち時間が顕著に見えていると推測できる。この待ち時間は 6.5 節でも説明した Nagle アルゴリズムのバッファリングによって引き起こされる。すなわち EventBuilder の実質的な処理時間は図 6.18(b) と図 6.18(c) に見られる分布のテールを差し引いたものである。

図 6.17 の分布のテールがリングバッファにデータが来るまでの待機時間であることがわかったので、その待機時間の長さについての考察を加える。この試験では、13 個あるリングバッファのそれぞれに次の 3 種類のデータが固定長で詰まるようになっている：

- 1 台の TTCFOB から来る 1 イベントあたり 20 バイトの TTC データ
- ダミーのトリガー判定情報を送信する 1 台の New SL から来る 1 イベントあたり 28 バイトのデータ
- ダミーデータを送信しない 11 台の New SL から来る 1 イベントあたり 12 バイトのデータ

New SL と TTCFOB の MSS は 1460 バイトに統一しているので、データが来るまでの待機時間が最も長いのはダミーデータを送信しない New SL のデータを保持するリングバッファを待つ場合である。すなわちリングバッファ



(a) 全 500 万イベントの処理時間の分布　　(b) 処理時間が 50 μs 未満のイベントの処理時間の分布

図 6.17　L1A レート 100 kHz における EventBuilder の全体処理時間の分布。SFO に記録されたイベントデータのうち、最初の 500 万イベントを解析した。図 6.17(a) は全 500 万イベントの処理時間の分布である。解析した 500 万イベント中、7 イベントだけ 2.3 ms から 4.1 ms の時間を要しており、これらはこのプロットから除いている。図 6.17(b) は図 6.17(a) の処理時間が 50 μs 未満の領域を拡大したものである。大半のイベントの処理時間が 3–5 μs 程度であるが、一部のイベントではリングバッファにデータが来るまで待つ処理が入るので、全体処理時間のテールは $O(1000\ \mu\text{s})$ まで伸びている。

にデータが揃うまでの平均的な待機時間は、

$$\frac{1}{100\ \text{kHz}} \times \frac{1460\ \text{bytes}}{12\ \text{bytes/event}} = 1.22\ \text{ms} \tag{6.6}$$

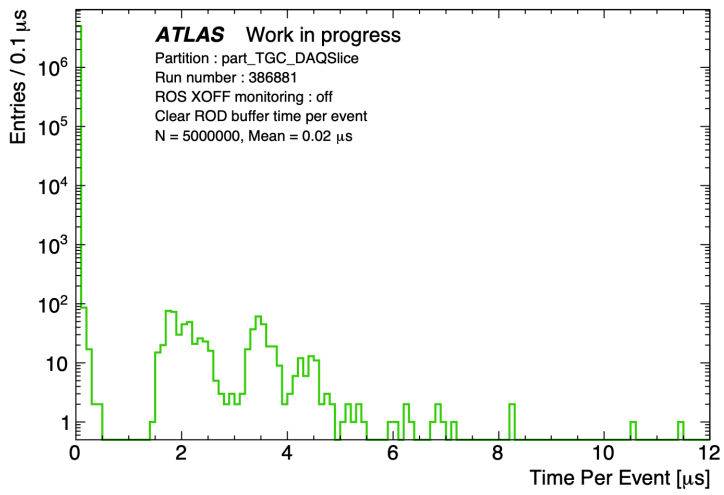
と計算される。しかし EventBuilder が行う処理の実質的な平均所要時間を考慮すると、実質的な待機時間は上記で計算された待機時間より短くなることが多い。この効果によって図 6.17 に見えているテールの分布は全体的に 1.22 ms よりも小さい方向にシフトしていると考えられる。

解析した 500 万イベントのうち、Nagle アルゴリズムの影響による所要時間を大きく超えるようなイベントが 7 つ観測された。これらのイベントは図 6.17 と図 6.18 のプロットを作る際には省いているが、それらの所要時間は 2.3 ms から 4.1 ms の範囲にあった。その 7 イベントのうち 4 イベントが Ring buffer to EB FIFO の処理で、3 イベントが Find data in EB FIFO の処理で時間を要していた。これらについては Collector アプリケーションにおける時間測定によって、Collector より上流で生じている遅延であることがわかっているが、その詳しい原因は現在も調べている最中である。

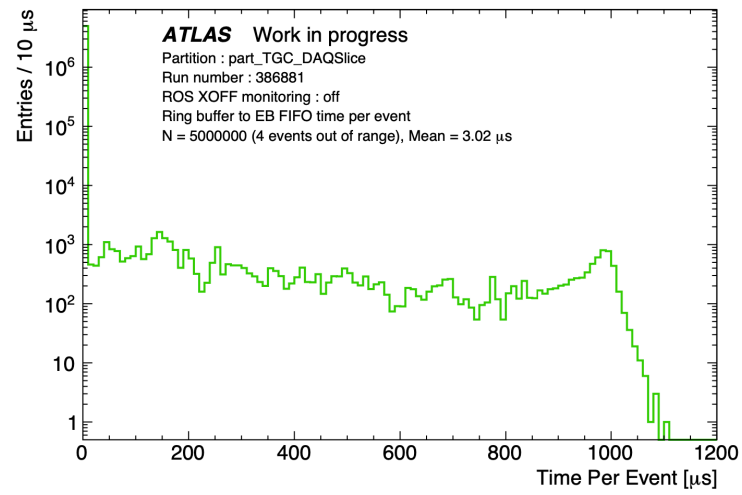
最後に、リングバッファにデータが来るまでの待ち時間の寄与を除いた実質的な処理時間を調べるために[*13]、L1A レートを 200 kHz にして全く同じ条件で時間測定を行なった[*14]。それぞれの L1A レートにおける各処理の平均所要時間を表 6.7 に示す。L1A レート以外の試験条件は変えていないため、リングバッファにデータが来るまでの待ち時間を含まない処理の所要時間はほとんど変わっていないことがわかる。対して、Ring buffer to EB FIFO と Find data in EB FIFO の所要時間は短くなっている。この比較から、L1A レートが 100 kHz の場合の全体処理時間のうち半分以上は、上記で説明したリングバッファにデータが来るまでの待ち時間の寄与によるものだと考えられる。すなわち今回の試験条件では、L1A レートが 100 kHz の場合においても、待ち時間を含まない EventBuilder の実質的な処理時間は 5 μs 以下であることが間接的にわかった。

[*13] EventBuilder のコードの関係上、リングバッファの待ち時間を分離して時間測定を行うのは難しかったので、このようにして間接的に処理時間を見積もることにした。

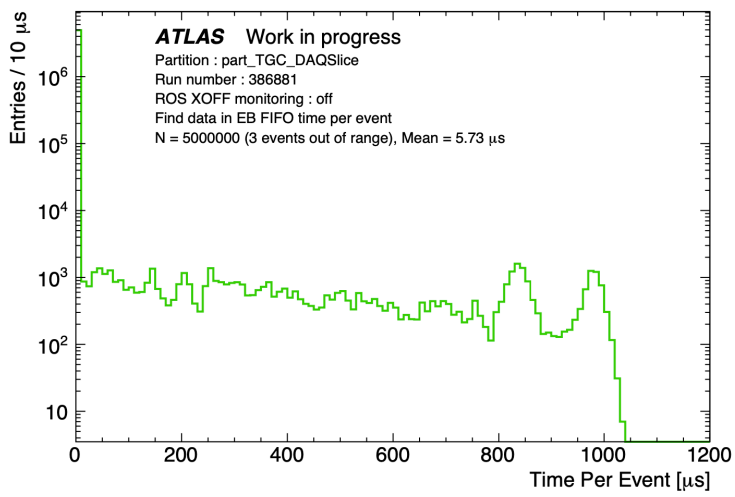
[*14] 厳密には試験時間だけ異なる。試験時間は 100 kHz の場合の半分の 45 分間であった。



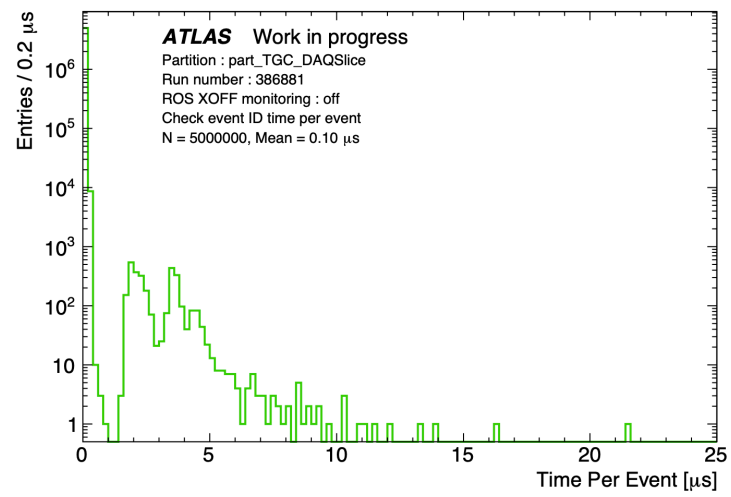
(a) Clear ROD buffer



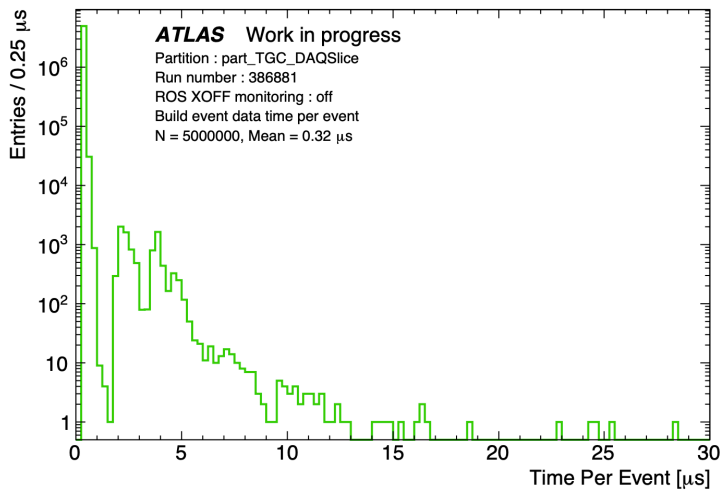
(b) Ring buffer to EB FIFO



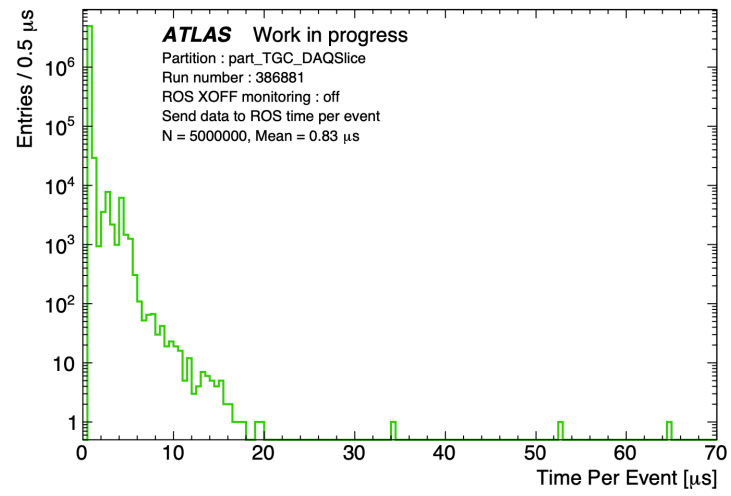
(c) Find data in EB FIFO



(d) Check event ID



(e) Build event data



(f) Send data to ROS

図 6.18　L1A レート 100 kHz における EventBuilder の各処理の所要時間の分布。Ring buffer to EB FIFO については 4 イベント、Find data in EB FIFO については 3 イベントだけ 2.3 ms から 4.1 ms の時間を要していたので、プロットからは除いた。 図 6.18(b) と図 6.18(c) の分布に見えるテールはリングバッファにデータが来るまでの待機時間の寄与によるものである。これらのテールを除いた処理時間が EventBuilder の実質的な処理時間である。

表 6.7　EventBuilder の各処理の平均所要時間。解析した 500 万イベントの平均値である。L1A レートを 200 kHz にした場合、Ring buffer to EB FIFO と Find data in EB FIFO の処理時間のみが大きく減少しているのは、リングバッファにデータが来るまでの待ち時間が短縮されたからである。

処理の名称	L1A 100 kHz の場合 [μs]	L1A 200 kHz の場合 [μs]
Clear ROD buffer	0.02	0.02
Ring buffer to EB FIFO	3.02	1.18
Find data in EB FIFO	5.73	2.61
Check event ID	0.10	0.11
Build event data	0.32	0.33
Send data to ROS	0.83	0.83
合計	10.0	5.07

EventBuilder の処理速度と ROS に送信するイベントデータの大きさとの関係

6.4.2 節で議論したように、Run 3 で実際に物理データを取得する際の SROD から ROS へのデータ送信量は、New SL におけるゼロサプレス圧縮率に依存する。余裕を持たせてゼロサプレス圧縮率を 10^{-2} と仮定した場合、SROD から ROS へのデータ送信量は 1 イベントあたり 1750 ビットになると計算された。そこで本研究で開発した読み出しシステムが Run 3 において安定して物理データを収集するために必要な条件を満たすかを調べるための試験を行なった。この試験では EventBuilder が 100 kHz で処理できるイベントデータの大きさの限界を調べるために、12 台の New SL のそれぞれから送信するダミーのトラック情報を増やし、EventBuilder が正常に動作するかを調べた。New SL から送信したトラック情報は MUCTPI に送信する形式のものと TGC BW から受信する形式のもので、これらのデータは EventBuilder にて図 6.14 に示したデータフォーマットにしたがってデコードされる。SFO に記録されたデータがデータフォーマットに沿っているかは確認している。

まず図 6.19 に、SROD から ROS に送信するイベントデータの大きさと EventBuilder の処理速度の関係を示す。EventBuilder の処理が間に合っている場合はその処理速度も L1A レートと同じ 100 kHz になっている。イベントデータの大きさが約 9000 ビットの場合までは EventBuilder の処理が 100 kHz に間に合っていることがわかる。6.4.2 節で大きめに見積もった Run 3 における平均イベントサイズは 1750 ビットであったので、SROD 単独では平均の大きさがその 5 倍程度のイベントデータまで busy 信号を出力せずに処理できることがわかった。

このことに加え、イベントデータの大きさが 9000 ビットを超えたあたりから EventBuilder の処理が追いつかなくなっていることがこのプロットから読み取れる。処理が間に合っていない場合はリングバッファの使用状況が 50% を超えてしまい、busy 信号が出力される。これによって L1A レートを抑制した状態で EventBuilder は正常に走り続ける。EventBuilder の処理速度の低下の原因を調べたところ、イベントデータの大きさの増加に伴って Build event data および Send data to ROS の処理に要する時間が大幅に増加していることがわかった。図 6.20 にそれらの所要時間が増加している様子を示す。イベントデータの大きさが 8160 ビットの場合、Build event data の所要時間は 4.8 μs で、Send data to ROS の所要時間は 1.8 μs であった。これらの値を表 6.7 で示した値と比較すると、この 2 つの処理だけで約 5.5 μs も所要時間が増加している。L1A レートが 100 kHz かつイベントデータの大きさが小さい場合は EventBuilder の処理時間に 5 μs 程度の余裕があったが、イベントデータが 9000 ビット程度になると Build event data と Send data to ROS の 2 つの処理の所要時間の増加によってこの余裕がなくなってしまうということが理解された。Build event data では主に New SL 用の 12 個の EB FIFO から ROD buffer にデータを移動させる処理を行なっているので、イベントデータの増加に伴って処理時間が増加することは想定通りである。Send data to ROS についても同様で、ROD buffer から S-LINK Card が持つバッファにデータを移動させ、ROS にデータを送信する処理を行うので、データ量の増加に応じて処理時間も増加する。以上のことから、L1A レートが 100 kHz の場合、EventBuilder および SROD ソフトウェアが busy 信号を出力せずに処理できるイベントデータの大きさの限界は約 9000 ビットであることが結論付けられた。これは要求されている 1750 ビットより十分に大

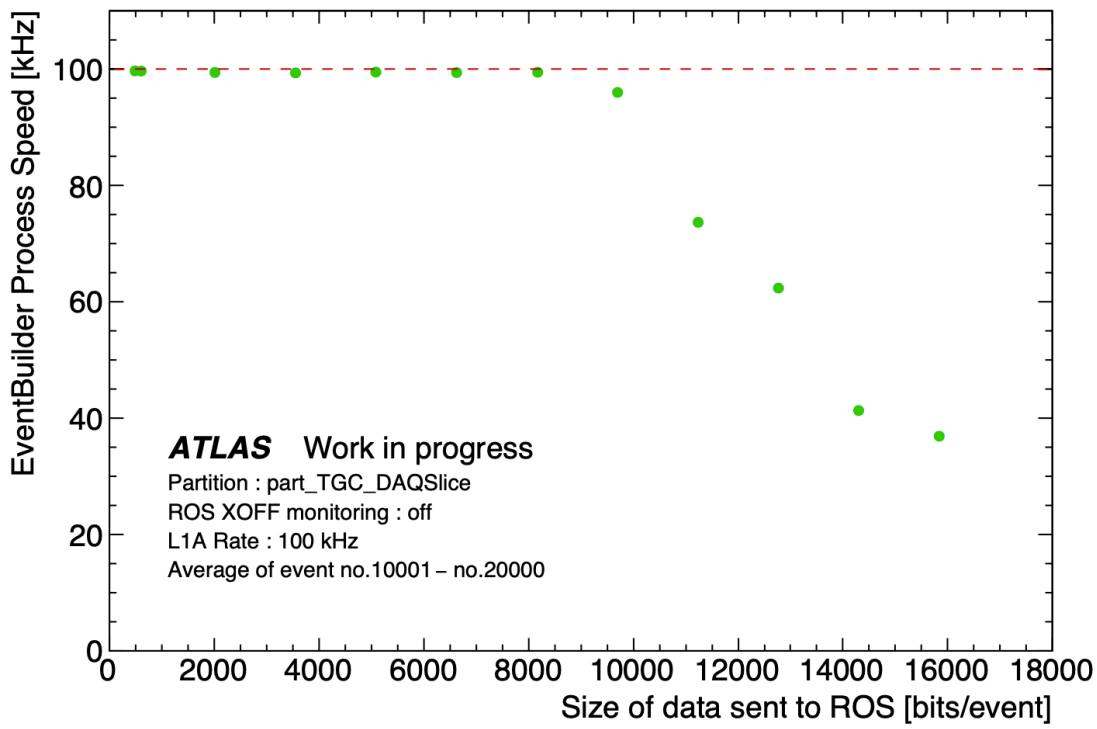
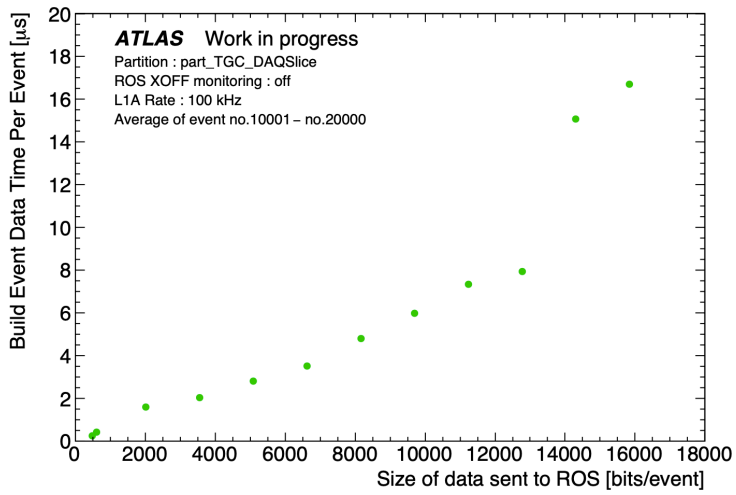


図 6.19　ROS に送信するイベントデータの大きさと EventBuilder の処理速度の関係。各データ点について EventBuilder が 5 分間以上正常に動作することを確認している。縦軸の EventBuilder の処理速度は、10001 イベント目から 20000 イベント目の計 10000 イベントを用いて算出した平均の処理速度である。L1A レートは 100 kHz に設定したので、EventBuilder の処理が間に合っている場合はその処理速度も 100 kHz になるはずである。ROS の XOFF を確認しない場合、100 kHz の速度で処理できるイベントデータの大きさは最大で約 9000 ビットであることがわかった。イベントデータの大きさが 9000 ビットを超えると EventBuilder の処理が間に合わず、リングバッファの使用状況が 50% を超え、busy 信号が出力された。

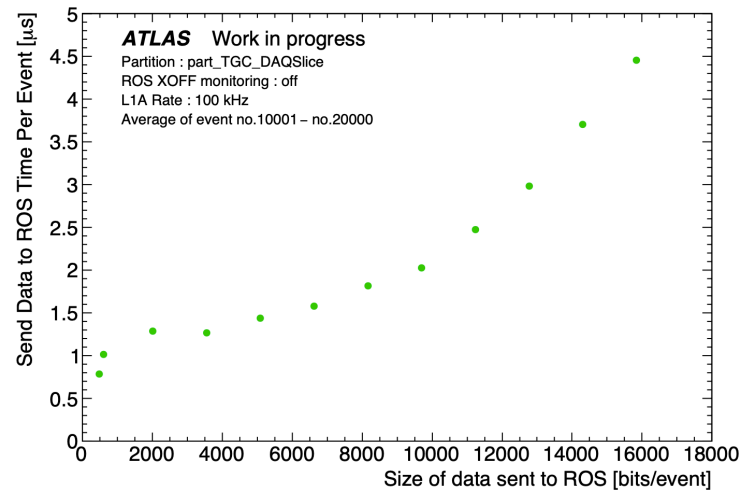
きいデータ量なので、必要な条件が満たされていることが確認できた。また、データ量が大きいときも busy 信号を出力しながら安定して運転が継続された。これは、データ量が大きいイベントが局在しても busy を活用することで、L1A レートを制御しながら安定して運転を継続できるシステムになっていることを示す結果である。

EventBuilder の処理速度と L1A レートの関係

最後に、イベントデータの大きさを固定して L1A レートを変化させて試験を行うことで、EventBuilder が正常にデータ処理を行えるような L1A レートを調べた。New SL からダミーのトラック情報を送ることで、SROD から ROS に送信するイベントデータの大きさが 608 ビットと 3552 ビットになるような 2 通りの条件で試験を行なった。図 6.21 にそれらの条件における EventBuilder の処理速度と L1A レートの関係を示す。イベントデータの大きさにかかわらず、L1A レートが 100 Hz から 100 kHz 以上までの広範囲の場合で EventBuilder が正常にデータ処理を行えることが確認できた。イベントデータの大きさが 608 ビットの場合は L1A レートが 200 kHz の場合まで busy 信号を出力せずにデータ処理を行えることが確認できた。L1A レートが 210 kHz の場合は busy 信号がわずかに出力された。イベントデータの大きさが 3552 ビットの場合は、180 kHz まで busy 信号が出力されなかった。L1A レートが 200 kHz および 210 kHz の場合はわずかに busy 信号を出力しながらもデータ処理を行うことができた。すなわち SROD 単独の性能として、Run 3 におけるイベントデータの平均的な大きさである 1750 ビットに対しても、L1A レートの観点からは十分な余裕があることがわかった。なお L1A レートを 220 kHz 以上にした場合はどちらのイベントデータの大きさにおいても、L1A が出力され始めてから 5 分以内に busy 信号が出力され続ける状態に陥り、測定を続けることができなくなった。これは EventBuilder の処理速度が L1A レートより十分遅くなったため生じた現象だと推測されるが、その正確な原因については今後も調査する予定である。

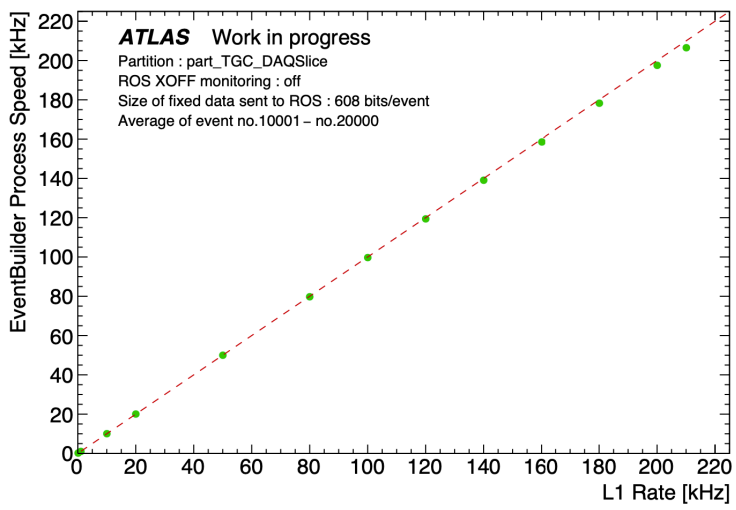


(a) Build event data の処理時間

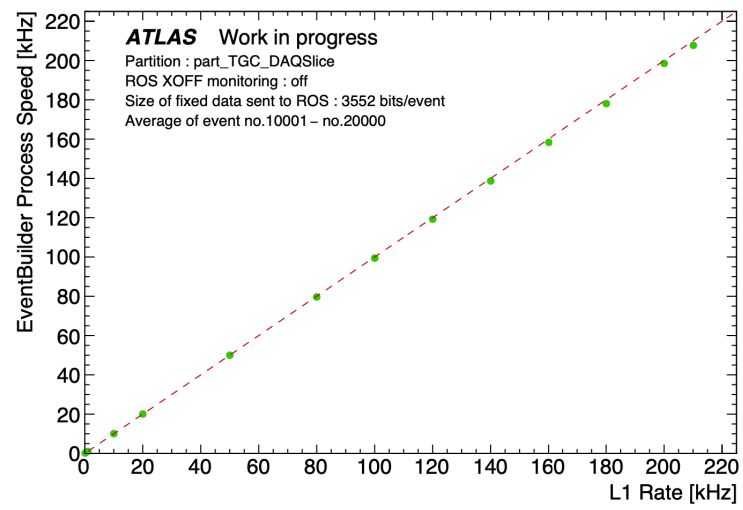


(b) Send data to ROS の処理時間

図 6.20　ROS に送信するイベントデータの大きさの増加に伴う処理時間の増加。図 6.20(a) は Build event data の所要時間の増加を、図 6.20(b) は Send data to ROS の所要時間の増加を表している。これらの所要時間は 10001 イベント目から 20000 イベント目の計 10000 イベントを用いて算出した平均の所要時間である。各データ点について EventBuilder が 5 分間以上正常に動作することを確認している。L1A レートは 100 kHz に設定した。



(a) イベントデータの大きさが 608 ビットの場合

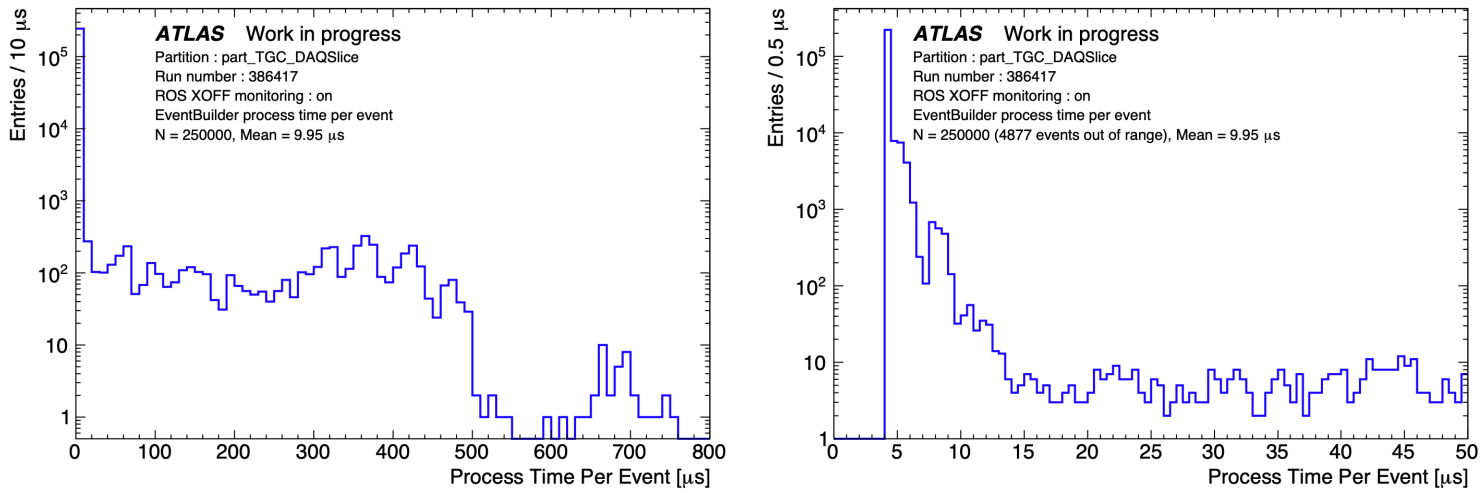


(b) イベントデータの大きさが 3552 ビットの場合

図 6.21　EventBuilder の処理速度と L1A レートの関係。横軸は OKS で設定した L1A の出力レート、縦軸は 10001 イベント目から 20000 イベント目までの計 10000 イベントを用いて算出した EventBuilder の平均処理速度である。各データ点について EventBuilder が 5 分間以上正常に動作することを確認している。EventBuilder が正常にデータを処理できているならばその処理速度は L1A レートとほぼ同じになるはずなので、赤い点線上にデータ点が乗る。イベントデータの大きさが、ゼロサプレス圧縮率を 10^{-2} と仮定した場合の Run 3 の平均値のおよそ 2 倍である 3552 ビットの場合でも、L1A レートが 180 kHz の場合まで正常にデータ処理が行えていた。すなわち L1A レートの観点からは十分に余裕があることが確認できた。

6.6.2　読み出しパス全体の性能

次に、S-LINK Card にて ROS から来る XOFF 信号を確認する場合での性能について議論する。ROS からの XOFF を正しく確認している場合は、ROS および SFO に正しくデータを送信できていることが保証されるので、読み出しパス全体の性能を評価することになる。



(a) 全 25 万イベントの処理時間の分布

(b) 処理時間が 50 μs 未満のイベントの処理時間の分布

図 6.22　L1A レート 100 kHz における EventBuilder の全体処理時間の分布。SFO に記録されたイベントデータのうち、最初の 25 万イベントを解析した。図 6.22(a) は全 25 万イベントの処理時間の分布である。図 6.22(b) は図 6.22(a) の処理時間が 50 μs 未満の領域を拡大したものである。大半のイベントの処理時間が 5 μs 程度であるが、一部のイベントではリングバッファにデータが来るまで待つ処理が入るので、全体処理時間のテールは $O(100\ \mu\text{s})$ まで伸びている。

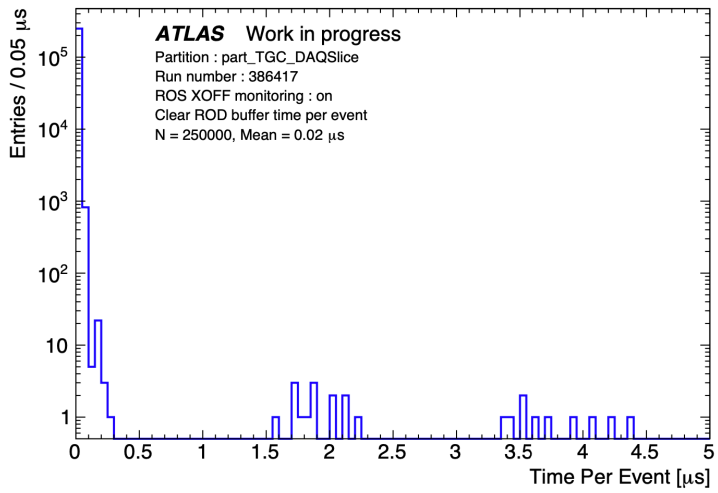
EventBuilder の処理時間の分布と内訳

まずは EventBuilder が行う 6 段階の処理の所要時間の分布と平均について調べた。時間測定の方法は 6.6.1 節の場合とほとんど同じであるが、使用した S-LINK Card のデバイスドライバーの性能の関係上[*15]、走らせた時間とサンプルしたイベント数が異なる。走らせた時間は 5 分程度で、SFO に記録されたイベントのうち、最初の 25 万イベントを解析した。

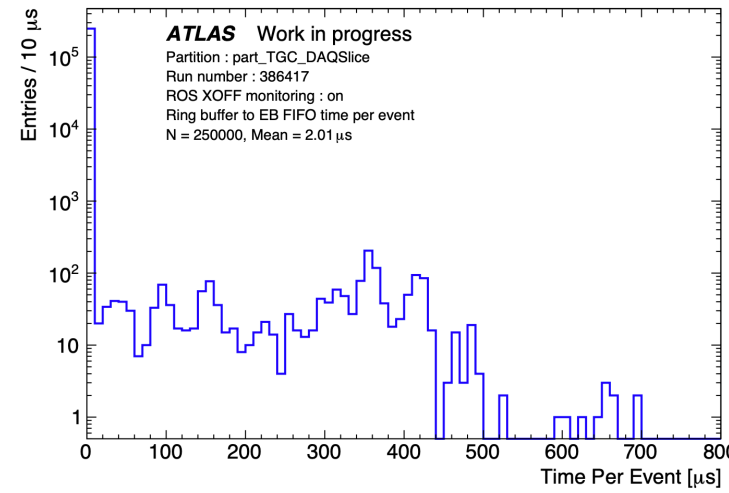
図 6.22 に EventBuilder の全体処理時間の分布を、図 6.23 に 6 つの処理のそれぞれの時間分布を示す。25 万イベントの平均処理時間は 9.95 μs で、100 kHz の L1A レートと整合した値になっている。図 6.22 の分布のテールは、図 6.23(b) および図 6.23(c) から、Ring buffer to EB FIFO および Find data in EB FIFO の処理に含まれるリングバッファにデータが来るまでの待ち時間によるものだと推測できる。しかし今回のテールは、ROS の XOFF を確認しない場合の時間分布におけるテール (図 6.17 参照) より短い。New SL および TTCFOB から送信している 1 イベントあたりのデータサイズは同じであるにもかかわらず、待ち時間は最大で 750 μs となっており、式 (6.6) で求めた値よりも短くなっている。この原因は、Send data to ROS の平均所要時間が広い分布を持たずに増加したことにある。まず図 6.23(f) を見ると、この分布は図 6.18(f) と似たような分布を持つが、その下限だけが右にシフトしている。これは ROS の XOFF を確認する処理の分だけ所要時間が延びるからである。次に表 6.8 に各処理の平均所要時間を示す。L1A レートを 100 kHz にした場合、ROS からの XOFF 信号を確認する場合の Send data to ROS の平均所要時間は 3.34 μs であった。これに対して、ROS からの XOFF を確認しない場合の Send data to ROS の平均所要時間は、表 6.7 に示したように 0.83 μs であった。すなわち ROS の XOFF を確認する処理によって Send data to ROS の平均所要時間が約 2.5 μs だけ増加する。この影響によりリングバッファにデータが来るまでの待機時間の分布は抑制されて広がりにくくなり、図 6.22 の分布のテールは式 (6.6) で求めた値に届きにくくなっていると考察できる。

ROS の XOFF を確認する場合においても、L1A レートを 200 kHz にして同様の時間測定を行なった。表 6.8 に各処理の平均所要時間を示す。L1A レートが 100 kHz の場合と比較すると、やはり Ring buffer to EB FIFO と Find data in EB FIFO の所要時間だけ大幅に短くなっている。すなわち ROS の XOFF を確認しない場合と同様

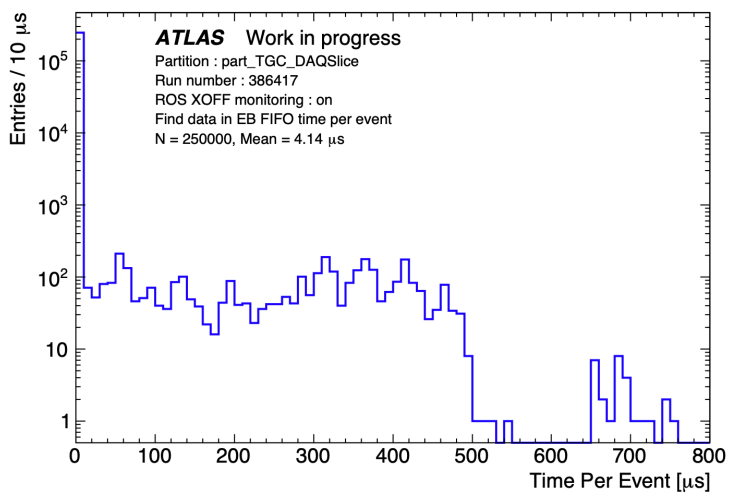
[*15] とある問題によって、この試験で使用した S-LINK Card のデバイスドライバーでは、ROS の XOFF を確認する場合に限って長時間走らせることが難しい。2021 年 1 月現在、この問題を解決するための開発が進められている。



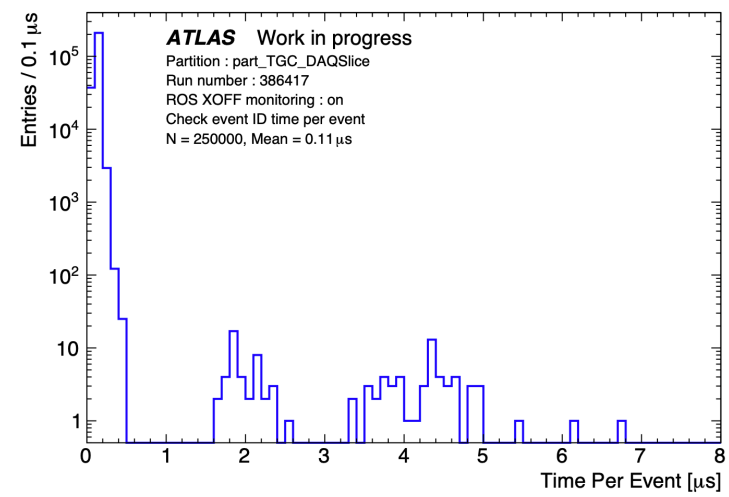
(a) Clear ROD buffer



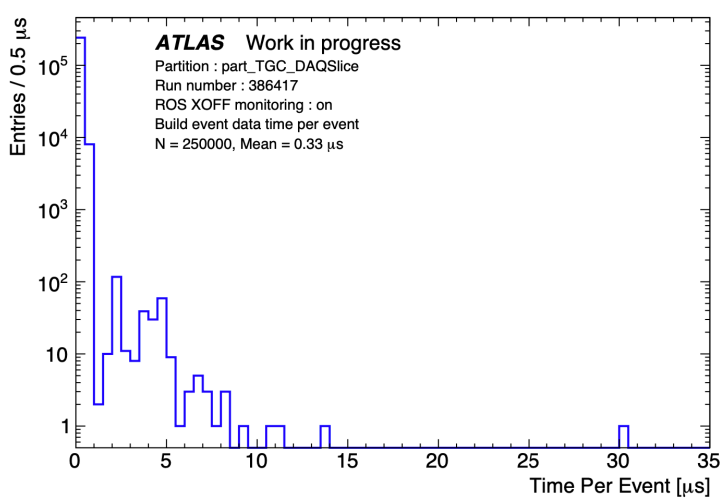
(b) Ring buffer to EB FIFO



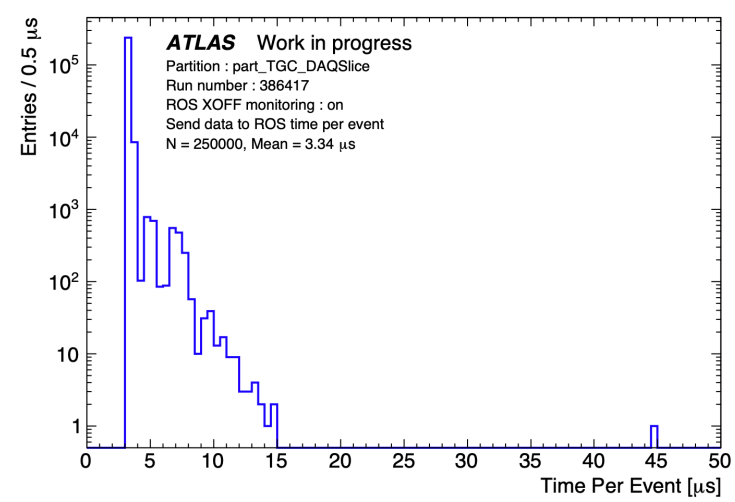
(c) Find data in EB FIFO



(d) Check event ID



(e) Build event data



(f) Send data to ROS

図 6.23　L1A レート 100 kHz における EventBuilder の各処理の所要時間の分布。SFO に記録されたイベントデータのうち、最初の 25 万イベントを解析した。

に、L1A レートが 100 kHz の場合は全体処理時間の 10 μs のうち少なくとも 5 μs 程度はデータが来るまでの待ち時間の寄与によるものである。今回の試験条件では ROS の XOFF を確認する場合においても、EventBuilder の処理時間にこれだけの余裕があることがわかった。これらの性能評価試験から得た処理時間の知見を参考に、Run 3 の物理ランにおいて想定されうる条件における読み出しパス全体の性能を評価していく。

表 6.8　EventBuilder の各処理の平均所要時間。解析した 25 万イベントの平均値である。

処理の名称	L1A 100 kHz の場合 [μs]	L1A 200 kHz の場合 [μs]
Clear ROD buffer	0.02	0.02
Ring buffer to EB FIFO	2.01	0.35
Find data in EB FIFO	4.14	0.95
Check event ID	0.11	0.10
Build event data	0.33	0.32
Send data to ROS	3.34	3.31
合計	9.95	5.06

EventBuilder の処理速度と ROS に送信するイベントデータの大きさとの関係

ROS から送られてくる XOFF 信号を確認しない場合と同じ方法で、New SL から送信するダミートラック情報を増やし、EventBuilder の処理速度と ROS に送信するイベントデータの大きさとの関係を調べた。まず図 6.24 に SROD から ROS に送信するイベントデータの大きさと EventBuilder の処理速度の関係を示す。イベントデータの大きさが 4832 ビット以下の場合は EventBuilder の処理が間に合っていたため、その処理速度は L1A レートと同じ 100 kHz になっていた。すなわち ROS の XOFF を確認して正しくデータを送信していることを保証した場合でも、Run 3 におけるイベントデータの平均サイズの予測値である 1750 ビットの 2.5 倍以上のデータ量まで安定して 100 kHz で処理・送信できることが確認できた。このことから、本研究で確立したトリガーデータ読み出しパスが十分な性能を持つことが示された。

イベントデータの大きさが約 5000 ビットを超えると、リングバッファの使用状況が 50% に達し始め、busy 信号が出力され始めた。5600 ビットまでは busy 信号を出しながら読み出しデータを処理することができていた。イベントデータの大きさに伴って busy 信号が出力される原因は EventBuilder の一部の処理において所要時間が増加するからである。ROS の XOFF を確認しない場合と同様に、イベントデータの増加に伴って Build event data と Send data to ROS の処理に要する時間が増加する。図 6.25 にそれらの処理の所要時間とイベントデータの大きさの関係を示す。これらの 2 つの処理の所要時間は、測定したイベントデータの大きさの範囲内では、イベントデータの大きさに応じてほぼ線形に増えることがわかる。特にイベントデータの大きさが約 5000 ビットの場合、Build event data の所要時間は 3 μs 程度で、Send data to ROS の所要時間は 5 μs 程度になっている。すなわちこれらの処理の所要時間を短縮することができれば、5000 ビット以上のイベントデータを安定して読み出すことができるようになる可能性がある。

Build event data の所要時間はデータの大きさにおおよそ比例していることが図 6.25(a) からわかる。比例関係を前提としてフィッティングを行なったところ、その結果は

$$(\text{Build event data time}) = (5.38 \pm 0.02) \times 10^{-4}\ \frac{\mu\text{s}}{\text{bits}} \cdot (\text{Size of data sent to ROS}) \tag{6.7}$$

となった。この結果の逆数を考えると、EventBuilder の他の処理を 1 μs 短縮することができれば、1 イベントあたりさらに約 1850 ビット大きいデータを送れるようになるということがわかる。この見積もりにより、これから行うさらなる性能改善に向けて定量的な目標を立てることが可能になった。

Send data to ROS の所要時間については、S-LINK 通信の最大通信速度と比較を行うために図 6.25(b) の赤線で

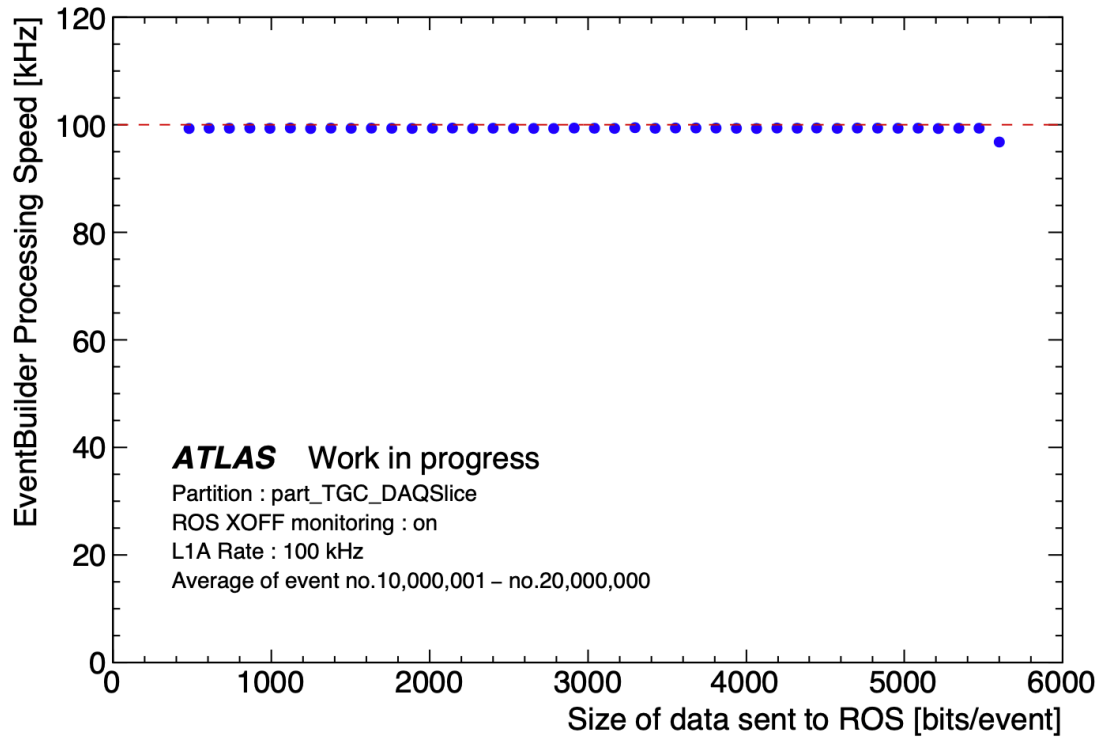
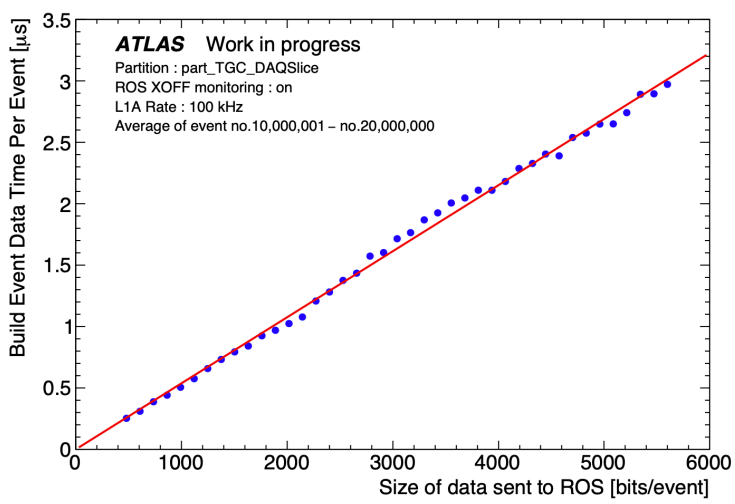
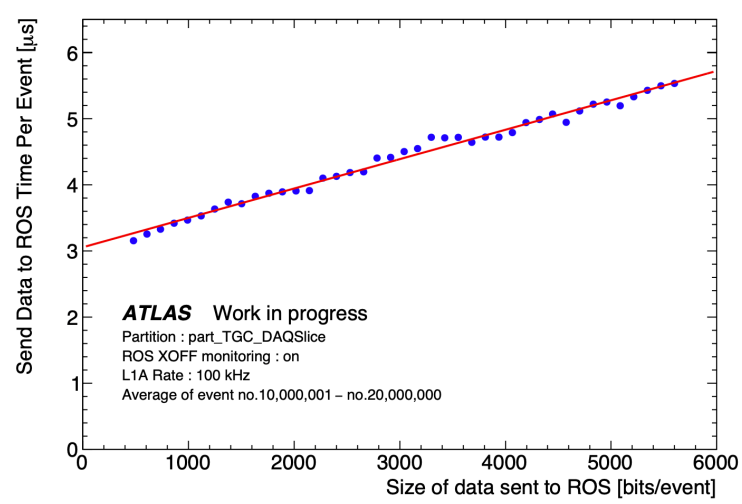


図 6.24　ROS に送信するイベントデータの大きさと EventBuilder の処理速度の関係。各データ点について EventBuilder が 5 分間以上正常に動作することを確認している。縦軸の EventBuilder の処理速度は、10,000,001 イベント目から 20,000,000 イベント目の計 10,000,000 イベントを用いて算出した平均の処理速度である。L1A レートは 100 kHz に設定したので、EventBuilder の処理が間に合っている場合はその処理速度も 100 kHz になる。ROS の XOFF を確認する場合、100 kHz の速度で処理できるイベントデータの大きさは最大で約 5000 ビットであることがわかった。イベントデータの大きさが 5000 ビットを超えると EventBuilder の処理が間に合わず、リングバッファの使用状況が 50% を超え、busy 信号が出力された。なおその busy 信号の出力は瞬間的なものだったため、5000 ビットを超える点の多くでも平均処理速度はおおよそ 100 kHz になっている。



(a) Build event data の処理時間



(b) Send data to ROS の処理時間

図 6.25　ROS に送信するイベントデータの大きさの増加に伴う処理時間の増加。図 6.25(a) は Build event data の所要時間の増加を、図 6.25(b) は Send data to ROS の所要時間の増加を表している。これらの所要時間は 10,000,001 イベント目から 20,000,000 イベント目の計 10,000,000 イベントを用いて算出した平均の所要時間である。各データ点について EventBuilder が 5 分間以上正常に動作することを確認している。L1A レートは 100 kHz に設定した。これらの処理の所要時間は 1 イベントあたりのデータサイズに応じてほぼ線形に増加することがわかる。

示すように線形フィッティングを行なった。その結果は、

$$(\text{Send data to ROS time}) = (4.44 \pm 0.07) \times 10^{-4}\ \frac{\mu\text{s}}{\text{bits}} \cdot (\text{Size of data sent to ROS}) + (3.06 \pm 0.02)\ \mu\text{s} \quad (6.8)$$

となった。1 次項の係数の逆数を考えることで、ROS へのデータ送信速度は (2.25 ± 0.04) Gbps と計算される。これは S-LINK 通信規格の最大通信速度である 2 Gbps とおおよそ合致しており [44]、SROD から ROS への実質的なデータ送信速度はハードウェアによって制限されていることが確認できた。また ROS の XOFF を確認しない場合の送信時間 (図 6.20(b) 参照) と比較することで、定数項の 3 μs の大半は ROS の XOFF を確認する処理に要している時間だと推測できる。よって Send data to ROS の所要時間は、ROS の XOFF を確認する処理を改善することで短縮できる可能性があることがわかった。

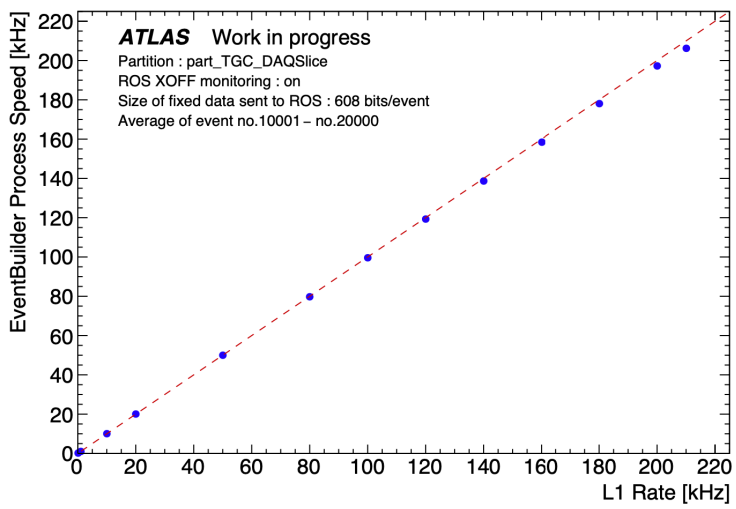
EventBuilder の処理速度と L1A レートの関係

最後に L1A レートの変化に対する EventBuilder の性能を評価した。New SL から送信するダミートラック情報を固定することで、SROD から ROS に送信するイベントデータの大きさを一定に保った。ROS からの XOFF を確認しない場合とは異なり、1 イベントあたりのデータサイズは 608 ビットと 1760 ビットの 2 通りの条件で試験した。6.4.2 節で議論したように、これらの値はゼロサプレス圧縮率を 10^{-3} および 10^{-2} と仮定した場合に Run 3 の物理ランで想定されるデータ量とおおよそ同じである。この 2 通りの条件における EventBuilder の処理速度と L1A レートの関係を図 6.26 に示す。イベントデータの大きさを 608 ビットとしたとき、L1A レートが 200 kHz の場合まで busy 信号を出力せずに読み出しデータを処理することができた。210 kHz ではわずかに busy を出力しながら走ることができたが、220 kHz 以上は busy 信号が常に出力される状態に陥り、測定ができなかった。この原因は 6.6.1 節にも前述した通り、EventBuilder の処理速度が L1A レートより十分遅くなったため生じたと考えられる。イベントデータのサイズが 1760 ビットの場合は、L1A レートが 170 kHz の場合まで busy 信号を出力せずにデータを処理し続けることができた。180 kHz 以上では EventBuilder の処理が間に合わず、常に busy 信号が出力される状態になった。ゼロサプレス圧縮率を大きめに 10^{-2} として見積もった大きさのイベントデータに対しても、100 kHz より十分速くトリガーデータを読み出すことができることが確認できた。すなわちこの性能評価試験によって、本研究で確立した SROD ソフトウェアを中心としたトリガーデータ読み出しパスの高レートに対する堅牢性が示された。

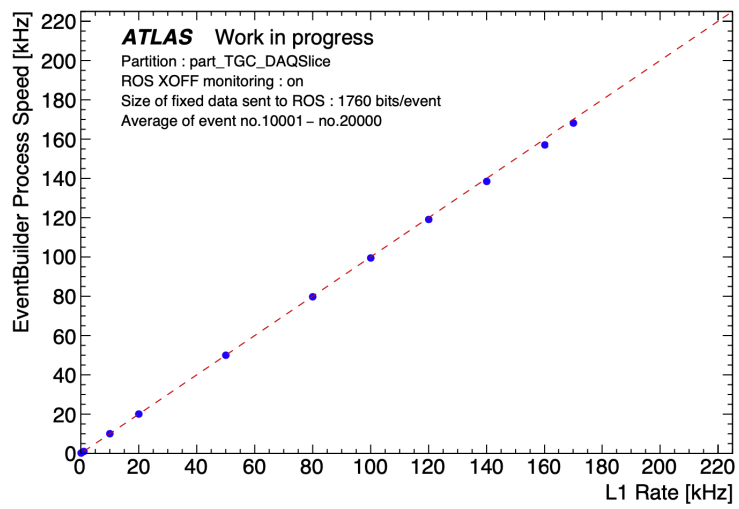
L1A レートを 100 kHz に固定してイベントデータを大きくした場合の最大データ処理速度は、5000 bits $\times$ 100 kHz = 500 Mbps であった。これに対し、イベントデータの大きさを 1760 ビットに固定して L1A レートを変化させた場合の最大データ処理速度は 1760 bits $\times$ 170 kHz = 299 Mbps となっている。さらに、イベントデータの大きさを 608 ビットに固定して L1A レートを変化させた場合の最大データ処理速度は 608 bits $\times$ 200 kHz = 122 Mbps となっている。よってデータ読み出しパスおよび SROD ソフトウェアの性能は、最大データ処理速度というパラメータを用いて一概に議論できないことがわかる。これはなぜかというと、図 6.25(b) に示すように、Send data to ROS の処理時間がイベントデータの大きさに比例していないからであると考察できる。イベントデータの大きさが小さくても Send data to ROS の処理時間は 3 μs ほど要するので、イベントの数が多いほど、すなわち L1A レートが高いほどオーバーヘッドが大きくなってしまう。これより、1 イベントあたりのデータサイズが大きいほど最大データ処理速度が大きくなる。実際の Run 3 の物理ランでは L1A レートは約 100 kHz に固定されていて、イベントデータの大きさのみが変動する。すなわちこのふるまいは SROD ソフトウェアおよびトリガーデータ読み出しパスにとっては都合が良い。このふるまいを念頭において、さらなる性能改善を行うことが可能であることがわかった。

6.7 SROD を用いた統合的なコミッショニングの現状

本研究で行なった SROD ソフトウェアの開発およびトリガーデータ読み出しパス全体の確立によって、New SL および TTCFOB の読み出しデータ、さらにはより上流にある TGC のフロントエンドエレクトロニクスから送られてくる信号に基づいた読み出しデータを確認することができるようになった。このトリガーデータ読み出しシス



(a) イベントデータの大きさが 608 ビットの場合



(b) イベントデータの大きさが 1760 ビットの場合

図 6.26　EventBuilder の処理速度と L1A レートの関係。横軸は OKS で設定した L1A の出力レート、縦軸は 10001 イベント目から 20000 イベント目までの計 10000 イベントを用いて算出した EventBuilder の平均処理速度である。各データ点について EventBuilder が 5 分間以上正常に動作することを確認している。EventBuilder が正常にデータを処理できているならばその処理速度は L1A レートとほぼ同じになるはずなので、赤い点線上にデータ点が乗る。イベントデータの大きさが、ゼロサプレス圧縮率を 10^{-2} と仮定した場合の Run 3 の平均値とほぼ同じである 1760 ビットの場合でも、L1A レートが 170 kHz の場合まで正常にデータ処理が行えていた。物理ランにおける L1A レートは 100 kHz なので、本研究で確立した読み出しパスが十分な性能を持つことが改めて確認できた。

テムを用いたコミッショニングが現在も進行している。本節ではこれらの統合的なコミッショニングの現状について簡単にまとめる。

6.7.1　Milestone Week における読み出しパスのバリデーション

2020 年 10 月 19 日から 23 日まで開催された第 4 回の Milestone Week にて、ATLAS Partition を用いて本研究で開発した SROD ソフトウェアを試験的に走らせ、確立した読み出しパスのバリデーションを行なった。SROD ソフトウェアを開発するために用いていた DAQSlice Partition では実物の HLT ではなく、Dummy HLT という HLT を模したプロセスによって ROS で受信したデータを SFO に記録していた。ATLAS Partition を用いて SFO にデータを記録することで、図 3.17 に示したような、実物の HLT を用いた読み出しパスのバリデーションができる。

このバリデーション試験では SROD A2 が 12 台の New SL と 1 台の TTCFOB から読み出しデータを受信するようなセットアップを用いた。それぞれの New SL からはトラック情報は送らず、L1ID や BCID を含む 12 バイトの情報のみを送った。L1A レートは約 100 Hz にして 5 分間ほど走らせ、SFO を経た Tier-0 にデータを記録した。記録されたデータを用いて作成したプロットを図 6.27 に示す。これは SROD A2 が読み出したイベントデータの BCID と、Run 2 でも使用されていた TGC ROD が読み出したイベントデータの BCID の差分を表す。このプロットから、本研究で確立したトリガーデータ読み出しパスが開通していることが、ATLAS Partition においても確認できた。

6.7.2　New SL および TTCFOB の BCR delay の調節

Run 3 から新たに導入する New SL や TTCFOB を正しく動作させるためには、それらが各イベントデータに付与する BCID が正しいもの (i.e. CTP が付与する BCID) になるようにしなければならない。New SL および TTCFOB は受信した BCR 信号を基に自身で BCID を数えるようになっている。すなわちこれらの新規エレクト

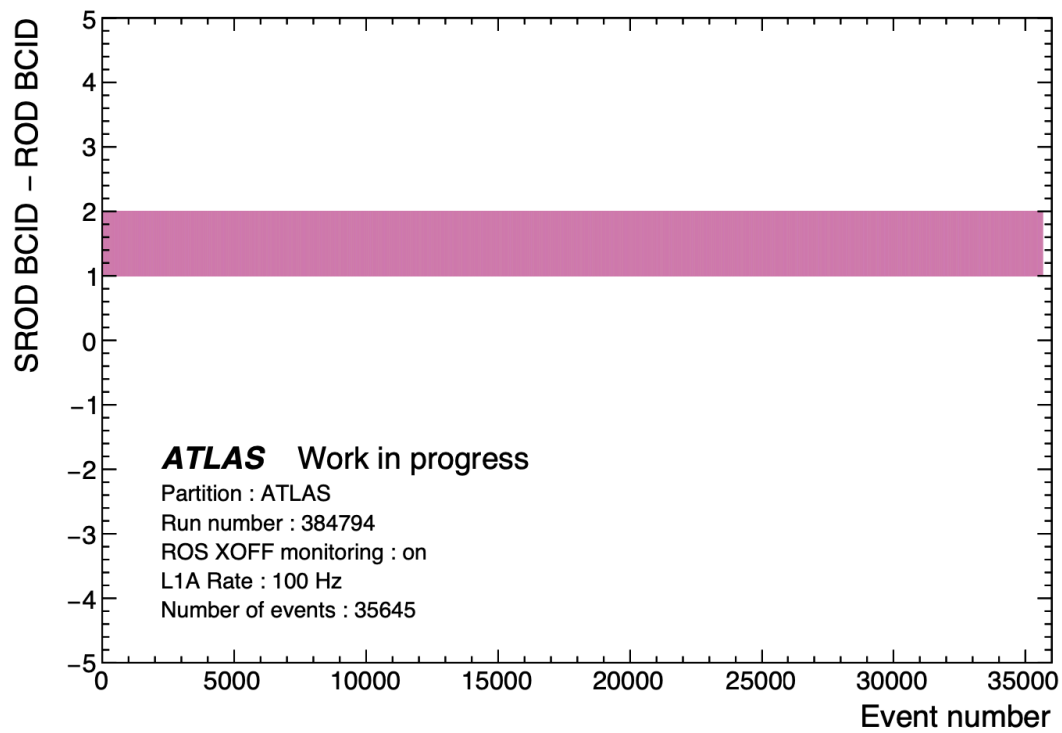


図 6.27　SROD と TGC ROD の読み出しデータの BCID の差分。ATLAS Partition を用いて L1A レート 100 Hz で、5 分間ほど SROD ソフトウェアおよび TGC ROD を用いてイベントデータを読み出した。Cern Data Centre である Tier-0 に記録された読み出しデータを解析することで作成した。6.7.2 節でも説明するように、このとき New SL および TTCFOB が付与していた BCID が TGC ROD が付与している BCID より 1 だけ大きいことがわかる。TGC ROD が付与している BCID は、Run 2 の時点で CTP が付与している BCID と同じになるように調節してある。

ロニクスが付与する BCID を合わせるためには、受信した BCR 信号に適切な遅延をかける必要がある。この遅延時間あるいはこの機構のことを BCR delay と呼ぶ。

SROD ソフトウェアの EventBuilder アプリケーションは、EB FIFO に入っているデータを確認する際に、12 台の New SL のそれぞれから来るイベントデータの BCID が TTCFOB から来る BCID と合致しているかを確認する。すなわち SROD ソフトウェアを用いることで、New SL および TTCFOB の間の BCID の不一致を検知することができる。しかし特に BCR delay を調節せずともこれらの不一致は検知されなかったので、はじめから 12 台の New SL と 1 台の TTCFOB の間の BCID は一致していた。しかしながらこれらの BCID が絶対的に正しいか、つまり CTP が付与する BCID と一致しているかはわからない。

そこで第 4 回の Milestone Week で、本研究で開発した SROD ソフトウェアを ATLAS Partition にて走らせ、Run 3 の新規エレクトロニクスが付与していた BCID が、CTP が付与する BCID と一致しているかを確認した。その結果、図 6.27 にすでに示したように、New SL や TTCFOB が付与していた BCID は CTP が付与する BCID よりも 1 だけ大きいことがわかった。すなわち New SL や TTCFOB において 1 BC 分の BCR delay をかける必要があることがわかった。現在はすべての New SL および TTCFOB に 1 BC 分の BCR delay をかけ、正しい BCID が付与されるようにしている。ATLAS Partition を用いて正しい BCID が付与できているかはまだ確認できていないが、図 6.28 に示すように、DAQSlice Partition を用いたバリデーションは済んでいる。

6.7.3　TGC のテストパルスを用いたトリガーパスおよびデータ読み出しパスの検証試験

これまでのトリガーデータ読み出しパスの開発では、主に TGC の TTC システムの A-side の LTP がランダムに出力した L1A 信号によってトリガーデータの読み出しを行なっていた。しかし実際の物理ランでは、ATLAS 実験室のフロントエンドエレクトロニクスのいずれかが送るヒット信号に基づいてトリガーを出力し、そのトリガー信

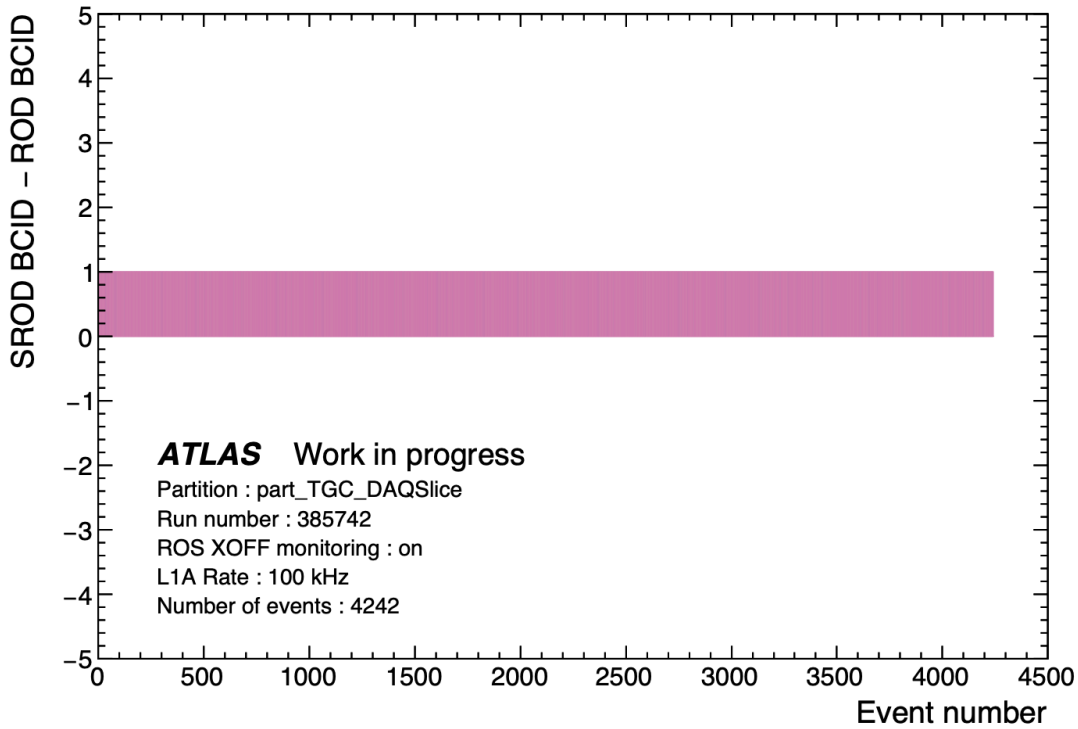


図 6.28　BCR delay 調節後の SROD と TGC ROD の読み出しデータの BCID の差分。DAQSlice Partition を用いて L1A レート 100 kHz で、数秒ほど SROD ソフトウェアおよび TGC ROD を用いてイベントデータを読み出した。SFO に記録された読み出しデータを解析することで作成した。New SL と TTCFOB の BCR delay を調節することで、SROD と TGC ROD が読み出すデータの BCID を合わせることができた。

号に基づいて CTP が出力した L1A によってトリガーデータを読み出す。すなわち実際にフロントエンドエレクトロニクスが出力したヒット信号に基づいてトリガーを出力し、そのトリガー信号に基づいた L1A によってトリガーデータを読み出すことで、フロントエンドエレクトロニクスからデータ記録までのすべてのパスが開通しているかを Run 3 の開始までに確認する必要がある。

2021 年 1 月現在は、ほとんどの New SL および G-Link Converter において、TGC BW および TGC EI/FI のヒット信号がすべての入力リンクにおいて同じタイミングで来るように調節済みである。しかし、Level-1 ミューオンエンドキャップトリガーのトリガー信号を CTP に渡す MUCTPI のハードウェアが USA15 にまだ設置されていないので、CTP は New SL が出力したトリガー信号に基づいて L1A を出力できるようにはまだなっていない。そこで現在の状況でトリガーパスおよびデータ読み出しパスの検証を行うために、2020 年 12 月 18 日に Level-1 ミューオンエンドキャップトリガーおよび TGC の中で閉じた試験を行なった。

図 6.29 にこの試験の模式図を示す。まず C02 セクターの TGC BW と TGC EI/FI の SLB ASIC から 11 kHz のテストパルス信号を打った。これらの信号は PS Board や HPT、G-Link Converter などを経て 1 台の New SL (C02-E03) に入力される。New SL では BW Coincidence と Inner Coincidence のロジックを通して、トリガー信号の出力が行われる。このトリガー信号を出力する先である MUCTPI はないので、代わりに L1A 信号を A-side の LTP に NIM 規格で送った。この L1A 信号は A-side の LTP から C-side の LTP および TTCFOB C1 を経て、その L1A を出力した New SL に返ってくる。そしてこの New SL は自身が出力した L1A 信号に基づいてトリガーデータを読み出す。トリガーデータの読み出しは本来は 4 BC 分行うが、今回は L1A に該当する 1 BC 分のデータのみを読み出すようにした。読み出されたトリガーデータは SROD C1 へと送られる。SROD C1 は New SL C02-E03 と TTCFOB C1 の読み出しデータを受信し、6.4 節で説明したフォーマットにしたがってデータをまとめ、ROS に送信する。最終的にそのデータは SFO に記録される。何度か SFO にデータを読み出し、L1A に対応する BC のデータが正しく読み出されるように New SL の L1 buffer の大きさを調節した。

最終的に SFO に記録されたデータを図 6.30 に示す。SROD C1 を用いて読み出した New SL C02-E03 のトリガーデータの部分を抜粋した。MUCTPI に送信するトリガーデータ、TGC EI/FI のヒットデータ、TGC BW

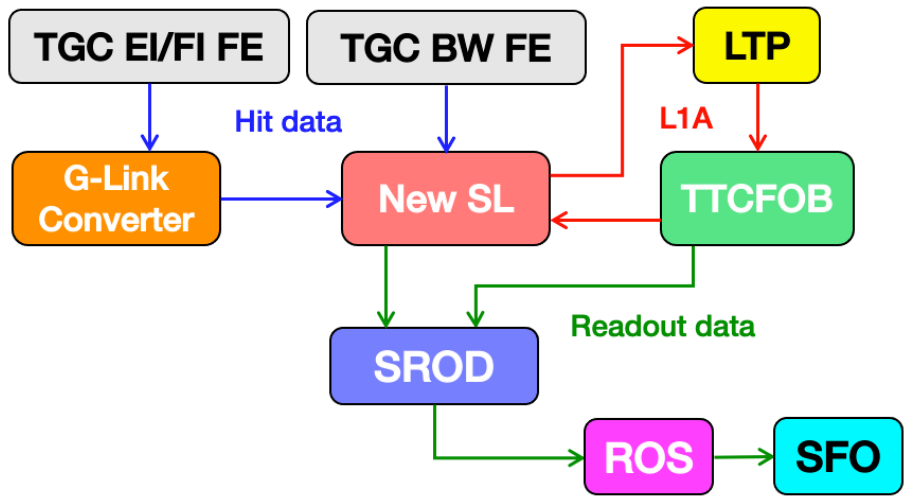


図 6.29　TGC のテストパルスを用いた検証試験のセットアップの模式図。

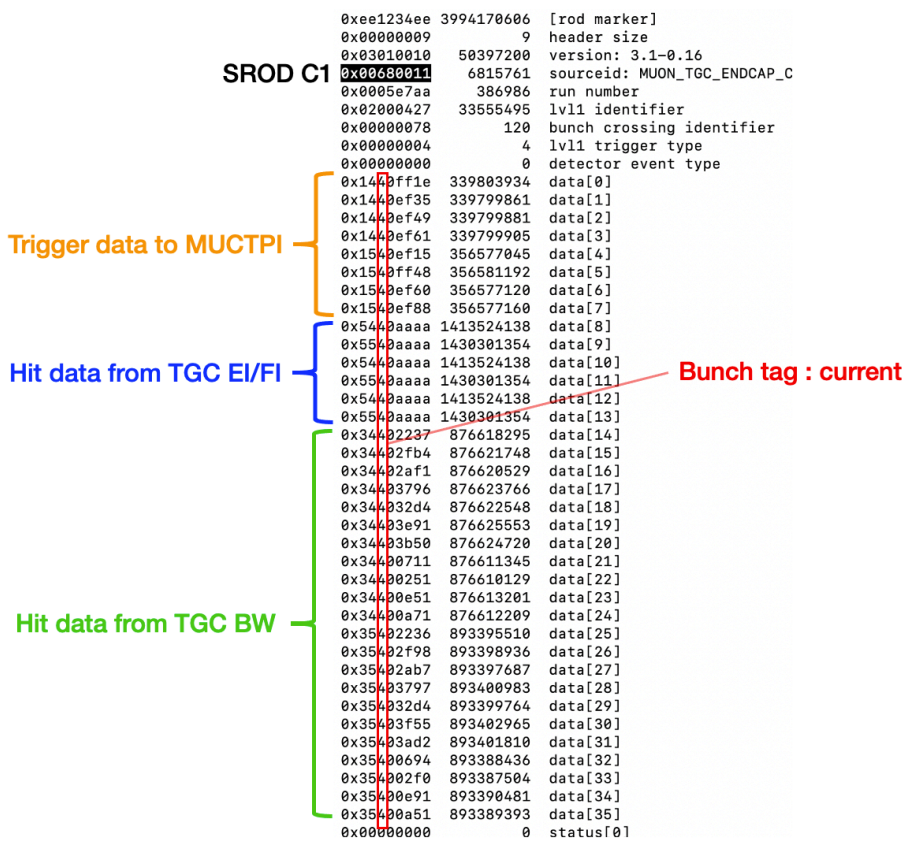


図 6.30　TGC のテストパルスを用いた検証試験において SFO に記録されたトリガーデータ。SROD C1 を用いて読み出した New SL C02-E03 のトリガーデータの部分を示している。データフォーマットは図 6.14 に従う。MUCTPI に送信するトリガーデータ、TGC EI/FI のヒットデータ、TGC BW のヒットデータがすべて正しく読み出されていることが確認できた。Bunch tag も current を示す 0x4 に揃っているので、すべて L1A に対応した BC のデータであることもわかる。テストパルスを用いた試運転によっても期待されるデータが正しく収集されていることが確認できた。

のヒットデータを図 6.14 と照らし合わせ、これらがすべてテストパルス信号と整合していることが確認された。Bunch tag が 0x4 に揃っていることから、L1A に対応した BC のデータが正しく読み出されていることも確認できた。この試験によって TGC BW と TGC EI/FI のフロントエンドエレクトロニクスから New SL のトリガーパス、New SL のトリガーロジックとトリガーデータ読み出し機構、SROD ソフトウェアにおけるデータフォーマットの実装、およびトリガーデータ読み出しパスの確立が実証された。

より詳細に渡る試験やデータ解析が 2021 年の課題となる。まず今回の試験では 1 台の New SL しか用いていないので、今後は複数の New SL を用いた検証試験を行う。また使用したヒット信号は TGC のものだけなので、い

ずれは NSW、RPC BIS78、および Tile Calorimeter のヒット信号を用いた試験を行う。そして MUCTPI のハードウェアがインストールされた後は CTP を用いた検証試験を行い、Run 3 に向けて L1 buffer の大きさの最終調節も行う。このようにして、Run 3 の開始までに確実に動作する Level-1 ミューオンエンドキャップトリガーシステムを構築する。

第 7 章

結論と今後の展望

2022 年開始予定の LHC-ATLAS 実験 Run 3 では、新しい検出器やエレクトロニクスを導入し、より質が高い物理データを収集する。特に陽子陽子衝突から生じるミューオンのヒット信号を基に事象選別を行う Level-1 ミューオンエンドキャップトリガーでは、偽のトリガー信号を削減するために NSW や RPC BIS78 などの新しい検出器を導入する。このため、New SL や TTCFOB などの新しいトリガーエレクトロニクスが開発された。またこの新しい Level-1 ミューオンエンドキャップトリガーシステムのトリガーロジックの性能評価をオフラインで行うために、SROD を中心としたソフトウェアベースのトリガーデータ読み出しシステムを新たに導入した。

本研究では全 72 台の New SL を中心とした Run 3 の新規エレクトロニクスを同期制御できるようなオンラインソフトウェアを開発し、新しい Level-1 ミューオンエンドキャップトリガーシステムおよびその中核をなすミューオントリガー検出器である TGC の検出器システムの統合制御を実現した。はじめに、ATLAS の TDAQ ソフトウェアを用いて新しい検出器エレクトロニクスを同期制御するために、それぞれに対応するオブジェクトを OKS データベースに定義し、それを用いて検出器システムの階層構造を定義した。次に、それぞれのエレクトロニクスの TTC 信号との同期や 72 台の New SL が持つ 1440 本の光リンクの確立を実現するために、システムのすべてのエレクトロニクスの同期を可能にする state transition を New SL クレートのエレクトロニクス、TGC のフロントエンドエレクトロニクス、TGC の TTC システムのそれぞれを制御するソフトウェアアプリケーションに実装した。これらの OKS やソフトウェアアプリケーションの開発によって、Run 3 の Level-1 ミューオンエンドキャップトリガーシステムが Run 2 でも使用していた TGC の検出器システムと同期して動作するような統合制御システムを構築した。さらに Grafana を用いたオンラインモニタリングを実現することで、検出器システムの誤動作やその兆候を検知しやすくし、システム全体の堅牢性を向上させた。この統合制御ソフトウェアの開発は、New SL などの新規エレクトロニクスを用いた試運転を可能にし、コミッショニングの進行に大きく貢献したという点で非常に重要な研究となった。

さらに本研究では SROD ソフトウェアの開発を行うことで、新しいトリガーデータ読み出しパスを確立した。まずは SROD の OKS やソフトウェアアプリケーションを開発することで、新たに USA15 に導入された全 6 台の PC 上でトリガーデータ読み出しソフトウェアが ATLAS の検出器システムの一部として走るような環境を構築した。また EventBuilder アプリケーションの性能を大幅に改善することで、100 kHz でのトリガーデータの読み出しを実現した。これに加えて、busy 信号を自動的に出力することで L1A レートを制御する機能を実装し、また pad word を導入することで L1A レートが低い場合に生じていた遅延時間を短縮する手法の新たに考案・実装した。最後に本研究で開発した SROD ソフトウェアおよび確立した読み出しパス全体の性能を初めて定量的に評価した。この性能評価によると、今回確立したトリガーデータ読み出しパスでは、100 kHz の L1A レートで Run 3 の物理ランにおけるイベントデータの平均的な大きさの 2.8 倍ものイベントデータを安定して読み出せることがわかった。これによって本研究で開発した SROD ソフトウェアおよび確立したトリガーデータ読み出しパスは Run 3 に向けて十分な性能を持つことを確認した。

今後は MUCTPI、NSW TP、および RPC BIS78 Pad Board などの New SL とデータ通信を行う新しいエレクトロニクスが USA15 に設置され、これらとのコミッショニングが本格的に開始される。また現在進行している

TGC の検出器システムや TMDB とのコミッショニングも引き続き行われる。最終的には Run 3 の開始までに、すべての検出器エレクトロニクスが同期して動作し、正しいタイミングでトリガーが出力されるような Level-1 ミューオンエンドキャップトリガーシステムを構築する必要がある。そのようなシステムの完成に向けて、本研究で開発した Level-1 ミューオンエンドキャップトリガーの統合制御ソフトウェアおよびトリガーデータ読み出しパスを最大限に活用するとともに、より洗練されたものに進化させていく。

謝辞

本研究を遂行するにあたって、多くの方々にお世話になりました。まず指導教員である奥村恭幸准教授には、研究内容や発表資料に関する数多くの助言、CERN 出張中の生活面のサポートなど、様々な面において助けていただきました。特にコードの書き方などテクニカルな面についても、何度も丁寧に相談に乗っていただいたおかげで、本研究をスムーズに進めることができました。また本研究に加え、自らの希望であった物理解析の研究を進めることに積極的に協力していただき、充実した修士課程を過ごすことができました。ありがとうございました。博士課程進学後もどうぞよろしくお願いいたします。

日々の研究に関して多くの助言をくださった、研究室ミーティングに参加してくださっているスタッフの皆さまにも心から感謝申し上げます。石野雅也教授には多くの点で大変お世話になりました。研究内容に関する的確な指摘や、共有した発表資料に対するすばやいフィードバックには毎度助けられています。齋藤智之助教にはオンラインソフトウェアの開発やコミッショニング全般、そして研究内容や発表資料に対する助言をいただき、とてもお世話になりました。増渕達也助教には主に SROD の開発でかなり助けていただきました。S-LINK Card のデバイスドライバのデバッグは引き続きお願いすることになるかと思います。今後ともよろしくお願いします。

他の ICEPP の皆さまにもお世話になりました。特に浅井祥仁センター長、澤田龍准教授、齊藤真彦特任助教には物理解析の研究について数多くの助言をいただきました。田中純一教授には CERN 出張中に Macbook が故障した際に別の Macbook を貸していただきすごく助かりました。他にも 2019 年の ICEPP 夏の学校、解析ミーティング、そして CERN での出張生活でも、多くの ICEPP のスタッフや学生の方々に助けていただきました。また秘書の皆さまにも出張手続きをはじめとした様々な面で支えていただきました。今後ともよろしくお願いいたします。

Run 3 に向けたアップグレード研究を共同で行なっている皆さまにも大変お世話になりました。青木雅人氏には TGC のオンラインソフトウェアや検出器システム全般についての指導、および CERN や KEK での研究面と生活面でとてもお世話になりました。前田順平氏にはオンラインソフトウェアや OKS の開発全般、特に SROD についてかなり深い部分まで教えていただきました。三野さん、忙しい中 SROD 開発を手伝っていただきありがとうございました。同期の辻川くん、New SL や TTCFOB のファームウェア開発とコミッショニングでいろいろ助けてくれてありがとう。奥村研究室の後輩の林くん、コミッショニングで収集したデータのデコードを手伝ってくれてありがとう。この他にも佐々木修氏、戸本誠氏、藏重久弥氏、隅田土詞氏、堀井泰之氏、水上さん、岡崎さん、竹田さん、日比さん、麻田さんには研究面および CERN や KEK 出張中の生活面で頻繁にお世話になりました。研究チームの皆さまに感謝申し上げます。今後ともよろしくお願いいたします。

そして ICEPP 同期の皆にもいろいろお世話になりました。田中、これからもよろしく。田中がフランス語に精通していて、しかもジュネーブに詳しかったおかげで初めての CERN 出張も心強かった。また近いうちにテニスしよう。仲間、単位とかお金遣いの面でなかなか破天荒で面白かったよ。2020 年度は CERN に行けなくて残念だったね。橋立さん、ICEPP 夏の学校でいろいろな場所に行けて良かったね。サレーブ山を爆速で駆け登る姿は未だに忘れられない。学部 4 年で同じ研究室だった島田、実験が違うと全然会わないね。ビールはほどほどに。米本くん、お花見の幹事をやってくれてありがとう。田中と島田と一緒に日帰りでベルンに行ったのは良い思い出だね。増田くん、M0 ゼミとかで結構理論の議論ができて楽しかったよ。またいろいろ議論しよう。山本くん、最後に話したのはいつかのオンライン飲み会だった気がする。卒業までまだ半年あるんだっけ？また会ったら話そう。

最後に、ここまで日々支えてくれた両親に感謝します。自由な道を選ばせてくれて本当にありがとうございます。これからもよろしくお願いします。

付録 A

理論の補足

ここでは主に第 2 章の議論で省略した事項を補う。

A.1　パウリ行列、ゲルマン行列、ガンマ行列

$SU(2)$ 群の生成子に用いられるパウリ行列 σ^i $(i=1,2,3)$ は

$$\sigma^1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma^2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma^3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{A.1}$$

として与えられる。また、$SU(3)$ 群の生成子の議論に現れるゲルマン行列 λ^α $(\alpha = 1, ..., 8)$ は次のように与えられる：

$$\begin{aligned} \lambda^1 &= \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \lambda^2 = \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \lambda^3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \lambda^4 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \\ \lambda^5 &= \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix}, \quad \lambda^6 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad \lambda^7 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix}, \quad \lambda^8 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix} \end{aligned} \tag{A.2}$$

最後に、フェルミオンの運動項 (i.e. ディラック方程式) に出てくるガンマ行列 γ^μ $(\mu = 0, 1, 2, 3)$ は、ディラック表現にしたがって、

$$\gamma^0 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad \gamma^i = \begin{pmatrix} 0 & \sigma^i \\ -\sigma^i & 0 \end{pmatrix} \tag{A.3}$$

として定義する。ただし式 (A.3) の 1 と 0 はそれぞれ 2 次の単位行列とゼロ行列を表す。

A.2　素粒子の標準模型の補足

素粒子の標準模型は電弱統一理論と量子色力学という 2 つのゲージ理論からなる理論体系である。A.2.1 節では電弱統一理論について、A.2.2 節では量子色力学について、量子補正を含まないツリーレベルの範疇で説明する。

A.2.1　電弱統一理論とヒッグス機構

1967 年に Weinberg と Salam らによって、電磁相互作用と弱い相互作用は 1 つの統一された理論として記述されることが示された。この理論は Weinberg–Salam 理論、あるいは電弱統一理論 (electroweak theory) と呼ばれ、電磁相互作用と弱い相互作用はまとめて電弱相互作用 (electroweak interaction) と呼ばれる。

電弱相互作用を記述する電弱統一理論は、非可換ゲージ群 $SU(2)$ と可換ゲージ群 $U(1)$ という 2 つのゲージ対称性を持つゲージ理論である。電弱統一理論に関わるこれらの特定のゲージ群は下添字をつけて、$SU(2)_L$ および

$U(1)_Y$ と表記される。すなわち、電弱統一理論は $SU(2)_L \times U(1)_Y$ ゲージ理論である。$SU(2)_L$ ゲージ群は粒子の左巻き成分にのみ作用する。すなわちフェルミオンの左巻き成分は

$$\begin{pmatrix} u_L \\ d_L \end{pmatrix}, \begin{pmatrix} c_L \\ s_L \end{pmatrix}, \begin{pmatrix} t_L \\ b_L \end{pmatrix}, \begin{pmatrix} \nu_{eL} \\ e_L \end{pmatrix}, \begin{pmatrix} \nu_{\mu L} \\ \mu_L \end{pmatrix}, \begin{pmatrix} \nu_{\tau L} \\ \tau_L \end{pmatrix} \tag{A.4}$$

という $SU(2)_L$ 二重項を構成する。他方で、フェルミオンの右巻き成分は $SU(2)_L$ ゲージ変換の下で不変なので、

$$u_R, c_R, t_R, d_R, s_R, b_R, e_R, \mu_R, \tau_R \tag{A.5}$$

という $SU(2)_L$ 一重項を構成する。また $U(1)_Y$ ゲージ群は、$U(1)_Y$ ゲージ群のチャージであるハイパーチャージ Y が 0 でないすべての粒子に作用する。

$SU(2)_L$ のゲージ場を W_μ とすると、これは一般的な $SU(2)$ 群の生成子 $T^a = \frac{\sigma^a}{2} (a = 1, 2, 3)$ および $SU(2)$ 群の 3 つのゲージ場成分 W^a_μ を用いて

$$W_\mu = W^a_\mu T^a = W^a_\mu \frac{\sigma^a}{2} \tag{A.6}$$

と書ける[*1]。ただし σ^a は A.1 節に記したパウリ行列である。同様に B_μ を $U(1)_Y$ のゲージ場とし、g_2 および g_Y を $SU(2)_L$ および $U(1)_Y$ ゲージ群のそれぞれに対応する結合定数とすると、電弱統一理論における共変微分は、

$$\begin{aligned} D^{EW(L)}_\mu &= \partial_\mu + i g_2 W^a_\mu \frac{\sigma^a}{2} + i g_Y B_\mu \frac{Y}{2} \\ D^{EW(R)}_\mu &= \partial_\mu + i g_Y B_\mu \frac{Y}{2} \end{aligned} \tag{A.7}$$

というように、$SU(2)_L$ 二重項に作用する $D^{EW(L)}_\mu$ と $SU(2)_L$ 一重項に作用する $D^{EW(R)}_\mu$ に分けて与えられる。以降では簡単のため、特に混乱が生じない場合はこれらはすべて D_μ と略記することにする。また 3 世代あることが知られているクォークやレプトンについても、世代数 $i = 1, 2, 3$ を用いて、

$$Q^i_L = \begin{pmatrix} u^i_L \\ d^i_L \end{pmatrix}, L^i_L = \begin{pmatrix} \nu^i_L \\ e^i_L \end{pmatrix}, u^i_R, d^i_R, e^i_R \tag{A.8}$$

と表記する。

上記の表記を用いると、電弱統一理論におけるラグランジアンの一部として、フェルミオンに関する項

$$\mathcal{L}_F = \overline{Q^i_L} i\gamma^\mu D_\mu Q^i_L + \overline{u^i_R} i\gamma^\mu D_\mu u^i_R + \overline{d^i_R} i\gamma^\mu D_\mu d^i_R + \overline{L^i_L} i\gamma^\mu D_\mu L^i_L + \overline{e^i_R} i\gamma^\mu D_\mu e^i_R \tag{A.9}$$

およびゲージ場に関する項

$$\mathcal{L}_G = -\frac{1}{4} W^a_{\mu\nu} W^{a\mu\nu} - \frac{1}{4} B_{\mu\nu} B^{\mu\nu} \tag{A.10}$$

が書ける。ただし、

$$\begin{aligned} W^a_{\mu\nu} &\equiv \partial_\mu W^a_\nu - \partial_\nu W^a_\mu - g_2 \varepsilon^{abc} W^b_\mu W^c_\nu \\ B_{\mu\nu} &\equiv \partial_\mu B_\nu - \partial_\nu B_\mu \end{aligned} \tag{A.11}$$

であり、γ^μ は A.1 節に記載してあるガンマ行列である。ε^{abc} は、

$$\left[\frac{\sigma^a}{2}, \frac{\sigma^b}{2} \right] = i\varepsilon^{abc} \frac{\sigma^c}{2} \tag{A.12}$$

によって定義される 3 階の完全反対称テンソルである。式 (A.9) および式 (A.10) を足し合わせたものは、任意のフェルミオンの左巻き成分 ψ_L および右巻き成分 ψ_R に対する $SU(2)_L \times U(1)_Y$ ゲージ変換

$$\begin{aligned} \psi_L \to \psi'_L &= e^{-i g_2 \Lambda^a(x) \frac{\sigma^a}{2} - i g_Y \Theta(x) \frac{Y}{2}} \psi_L \\ \psi_R \to \psi'_R &= e^{-i g_Y \Theta(x) \frac{Y}{2}} \psi_R \end{aligned} \tag{A.13}$$

[*1] 繰り返される添字については和をとり、以後もそうする。

の下で不変であることが確認できる。ただし、$\Lambda^a(x)$ および $\Theta(x)$ はそれぞれ $SU(2)_L$ および $U(1)_Y$ ゲージ変換の局所的な位相である。

しかしながら式 (A.9) および式 (A.10) によって与えられるフェルミオンとゲージ場のラグランジアンだけでは、フェルミオンやゲージボソンに質量を与えることはできない。ラグランジアンが持つゲージ対称性をあらわに破らずにそれらの粒子に質量を与えるためには、以下に説明するヒッグス機構が必要になる。

電弱統一理論にヒッグス機構を組み込むには、新たに $Y=1$ を持つ $SU(2)_L$ 二重項として

$$\Phi = \begin{pmatrix} \phi^+ \\ \phi^0 \end{pmatrix} \equiv \frac{1}{\sqrt{2}} \begin{pmatrix} \phi_1 + i\phi_2 \\ \phi_3 + i\phi_4 \end{pmatrix} \tag{A.14}$$

という形をしたヒッグス二重項 Φ を導入しなければならない。ここで ϕ^+ および ϕ^0 は複素スカラー場であり、上添字の “+” と “0” はそれぞれが持つ電荷を表す。このヒッグス二重項 Φ を使って、ラグランジアン

$$\mathcal{L}_H = (D_\mu \Phi)^\dagger (D^\mu \Phi) - V(\Phi) \tag{A.15}$$

を考える。ここでヒッグスポテンシャル $V(\Phi)$ は 2 つの実パラメータ $\mu^2 < 0$ と $\lambda > 0$ を用いて

$$V(\Phi) = \mu^2 \Phi^\dagger \Phi + \lambda (\Phi^\dagger \Phi)^2 \tag{A.16}$$

で与えられる。μ^2 が負であるので、$V(\Phi)$ の最小点では

$$\Phi^\dagger \Phi = \frac{1}{2}(\phi_1^2 + \phi_2^2 + \phi_3^2 + \phi_4^2) = -\frac{\mu^2}{2\lambda} \tag{A.17}$$

となる。この最小点を満たす状態を “真空状態” と呼び、そのときのヒッグス二重項の期待値を “真空期待値” と呼ぶ。例えば、

$$\phi_1 = \phi_2 = \phi_4 = 0, \quad \phi_3 = \sqrt{-\frac{\mu^2}{\lambda}} \equiv v \tag{A.18}$$

とすれば真空状態が実現される。このときの真空期待値 $\langle \Phi \rangle$ は式 (A.14) より

$$\langle \Phi \rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v \end{pmatrix} \tag{A.19}$$

である。また、ヒッグス二重項 Φ は真空期待値の近傍で近似的に

$$\Phi(x) \sim \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ v + h(x) \end{pmatrix} \tag{A.20}$$

と表すことができる。ここで $h(x)$ はヒッグス場という実スカラー場である。ゲージボソンが持つ質量は、式 (A.19) を式 (A.15) の第一項に代入することで得られる。すなわち、

$$\begin{aligned} |D_\mu \langle \Phi \rangle|^2 &= \frac{1}{8} v^2 g_2^2 \left(W_\mu^1 W^{1\mu} + W_\mu^2 W^{2\mu} \right) + \frac{1}{8} v^2 (g_Y B_\mu - g_2 W_\mu^3)(g_Y B^\mu - g_2 W^{3\mu}) \\ &= \left(\frac{1}{2} v g_2 \right)^2 W_\mu^+ W^{-\mu} + \frac{1}{2} \left(\frac{1}{2} v \sqrt{g_Y^2 + g_2^2} \right)^2 Z_\mu Z^\mu + \frac{1}{2} \cdot 0^2 A_\mu A^\mu \end{aligned} \tag{A.21}$$

となる。2 行目では、$W_\mu^\pm, Z_\mu, A_\mu$ を

$$W_\mu^\pm \equiv \frac{1}{\sqrt{2}} (W_\mu^1 \mp i W_\mu^2), \qquad Z_\mu \equiv \frac{g_2 W_\mu^3 - g_Y B_\mu}{\sqrt{g_Y^2 + g_2^2}}, \qquad A_\mu \equiv \frac{g_Y W_\mu^3 + g_2 B_\mu}{\sqrt{g_Y^2 + g_2^2}} \tag{A.22}$$

として定義した。$W_\mu^\pm$ は W ボソンを、Z_μ は Z ボソンを、そして A_μ は光子を表す。それらの質量は式 (A.21) より、

$$M_W = \frac{1}{2} v g_2, \qquad M_Z = \frac{1}{2} v \sqrt{g_Y^2 + g_2^2}, \qquad M_A = 0 \tag{A.23}$$

となることがわかる。つまり、W ボソンおよび Z ボソンは新たに導入したヒッグス二重項の 4 つの自由度のうちの 3 つを吸収し、質量を獲得する。残る 1 つの自由度はヒッグス場 $h(x)$ が保持しており、これが 2012 年に ATLAS および CMS 実験によって発見されたヒッグスボソンの正体である。ヒッグスボソンの質量 m_h は式 (A.20) を式 (A.16) に代入することで、

$$m_h = \sqrt{2\lambda}v \tag{A.24}$$

となることがわかる。

このようにヒッグス機構を導入することで、ヒッグスポテンシャルの最小点近傍では、系のゲージ対称性をあらわに破らずにゲージボソンに質量を持たせることができた。すなわち低エネルギー領域では $SU(2)_L \times U(1)_Y$ ゲージ対称性が自発的に破れ、電磁相互作用に関する $U(1)_{em}$ ゲージ対称性のみが残る。

同じヒッグス二重項 Φ を使ってフェルミオンにも質量を与えることができる。ただし議論を簡単にするために、以降では質量固有状態と弱固有状態の混合を議論するのに必要な CKM (Cabibbo–Kobayashi–Maskawa) 行列や PMNS (Pontecorvo–Maki–Nakagawa–Sakata) 行列は省略する[*2]。また標準模型においてニュートリノは質量を持たないとされてきたので、ここでもニュートリノの質量は 0 であると仮定する[*3]。

荷電レプトンに質量を与えるには、ラグランジアンに湯川相互作用項

$$-y_L^{ij}\left\{\overline{L_L^{0i}}\Phi e_R^{0j} + \overline{e_R^{0i}}\Phi^\dagger L_L^{0j}\right\} \tag{A.25}$$

を加える。ここで y_L^{ij} はレプトンに関する第 i 世代と第 j 世代間の湯川結合定数である。またレプトン場の上添字の 0 は、それらが弱固有状態であることを表す。低エネルギー領域におけるフェルミオンの質量を求めるには、式 (A.19) で与えたヒッグス二重項の真空期待値を式 (A.25) に代入すれば良い。すると、

$$\begin{aligned}-y_L^{ij}\left\{\overline{L_L^{0i}}\langle\Phi\rangle e_R^{0j} + \overline{e_R^{0i}}\langle\Phi^\dagger\rangle L_L^{0j}\right\} &= -\frac{y_L^{ij}v}{\sqrt{2}}\left(\overline{e_L^{0i}}e_R^{0j} + \overline{e_R^{0i}}e_L^{0j}\right)\\ &= -\frac{y_L^{'i}v}{\sqrt{2}}\left(\overline{e_L^{i}}e_R^{i} + \overline{e_R^{i}}e_L^{i}\right)\end{aligned} \tag{A.26}$$

となる。ただし 2 行目では弱固有状態から質量固有状態に移行し、このときの湯川結合定数を $y_L^{'i}$ とおいた。結局、第 i 世代の荷電レプトンの質量 m_e^i は

$$m_e^i = \frac{y_L^{'i}v}{\sqrt{2}} \tag{A.27}$$

で与えられることがわかった。

ダウンタイプのクォーク $d^i = d, s, b$ については、荷電レプトンと同じく左巻き成分が $SU(2)_L$ 二重項の下成分に属するため、全く同じ方法で質量を与えることができる。しかしながら、アップタイプのクォーク $u^i = u, c, t$ に質量を与えるためにはやや工夫が必要である。そのためにヒッグス二重項 Φ を用いて、

$$\widetilde{\Phi} \equiv i\sigma^2\Phi^* = \begin{pmatrix}(\phi^0)^*\\ -\phi^-\end{pmatrix} \sim \frac{1}{\sqrt{2}}\begin{pmatrix}v+h\\ 0\end{pmatrix} \tag{A.28}$$

を事前に定義しておく。この $\widetilde{\Phi}$ を用いることで、クォークの湯川相互作用項は

$$-y_d^{ij}\left\{\overline{Q_L^{0i}}\Phi d_R^{0j} + \overline{d_R^{0i}}\Phi^\dagger Q_L^{0j}\right\} - y_u^{ij}\left\{\overline{Q_L^{0i}}\widetilde{\Phi} u_R^{0j} + \overline{u_R^{0i}}\widetilde{\Phi}^\dagger Q_L^{0j}\right\} \tag{A.29}$$

と書ける。y_u^{ij} および y_d^{ij} は第 i 世代と第 j 世代間のアップおよびダウンタイプクォークの湯川結合定数である。レプトンのときと同様に、Φ および $\widetilde{\Phi}$ の真空期待値を式 (A.29) に代入することでクォークの質量項が得られる。

---

*2 CKM 行列および PMNS 行列はそれぞれクォークおよびレプトンの質量固有状態と弱固有状態を結びつける 3×3 の変換行列である。

*3 ニュートリノ振動の観測によってニュートリノは質量を持つことがわかっているが、どのようにして質量を獲得するかはまだ解明されていない。

実際、

$$
\begin{aligned}
&-y_d^{ij}\left\{\overline{Q_L^{0i}}\langle\Phi\rangle d_R^{0j}+\overline{d_R^{0i}}\langle\Phi^\dagger\rangle Q_L^{0j}\right\}-y_u^{ij}\left\{\overline{Q_L^{0i}}\langle\widetilde{\Phi}\rangle u_R^{0j}+\overline{u_R^{0i}}\langle\widetilde{\Phi}^\dagger\rangle Q_L^{0j}\right\}\\
&=-\frac{y_d^{ij}v}{\sqrt{2}}\left(\overline{d_L^{0i}}d_R^{0j}+\overline{d_R^{0i}}d_L^{0j}\right)-\frac{y_u^{ij}v}{\sqrt{2}}\left(\overline{u_L^{0i}}u_R^{0j}+\overline{u_R^{0i}}u_L^{0j}\right)\\
&=-\frac{y_d^{'i}v}{\sqrt{2}}\left(\overline{d_L^{i}}d_R^{i}+\overline{d_R^{i}}d_L^{i}\right)-\frac{y_u^{'i}v}{\sqrt{2}}\left(\overline{u_L^{i}}u_R^{i}+\overline{u_R^{i}}u_L^{i}\right)
\end{aligned}
\tag{A.30}
$$

より、第 i 世代のアップおよびダウンタイプクォークの質量 m_u^i および m_d^i は、

$$
m_{u,d}^i=\frac{y_{u,d}^{'i}v}{\sqrt{2}} \tag{A.31}
$$

で与えられる。ただし、$y_{u,d}^{'i}$ は第 i 世代のアップおよびダウンタイプクォークの質量固有状態における湯川結合定数である。まとめると、電弱統一理論における湯川相互作用項 $\mathcal{L}_Y$ は、

$$
\mathcal{L}_Y=-y_L^{ij}\left\{\overline{L_L^{0i}}\Phi e_R^{0j}+\overline{e_R^{0i}}\Phi^\dagger L_L^{0j}\right\}-y_d^{ij}\left\{\overline{Q_L^{0i}}\Phi d_R^{0j}+\overline{d_R^{0i}}\Phi^\dagger Q_L^{0j}\right\}-y_u^{ij}\left\{\overline{Q_L^{0i}}\widetilde{\Phi}u_R^{0j}+\overline{u_R^{0i}}\widetilde{\Phi}^\dagger Q_L^{0j}\right\} \tag{A.32}
$$

である。

ゆえに電弱統一理論の最終的なラグランジアン $\mathcal{L}_{EW}$ は、フェルミオン項 (A.9)、ゲージ項 (A.10)、ヒッグス項 (A.15)、そして湯川相互作用項 (A.32) をすべて足し合わせることで、

$$
\mathcal{L}_{EW}=\mathcal{L}_F+\mathcal{L}_G+\mathcal{L}_H+\mathcal{L}_Y \tag{A.33}
$$

となる。

A.2.2 量子色力学

強い相互作用を記述する量子色力学 (Quantum Chromodynamics, QCD) は 1950–60 年代に理論として徐々に発展し、1970 年代前半にそのくりこみ可能性や漸近的自由性が証明されたことで、理論として注目され始めた。強い相互作用はその漸近的自由性により、低エネルギー領域ではその結合定数が大きくなってしまうので、摂動計算が適用できない。そのため QCD において量子効果を考えるには格子ゲージ理論が使われ、今現在も多くの研究が進展している。本小節ではそういった複雑な量子効果には触れず、あくまでツリーレベルの範疇の議論をまとめる。

強い相互作用を記述する QCD は $SU(3)$ ゲージ理論という非可換ゲージ理論である。強い相互作用を表す特定の $SU(3)$ ゲージ群は $SU(3)_c$ と表記され、強い相互作用をするクォークは $SU(3)_c$ 三重項を形成する。$SU(3)_c$ ゲージ対称性の自由度は “カラー” と呼称され、それらを三原色に擬えて赤 (red)、緑 (green)、青 (blue) と呼ぶことがある。ここでは簡単のためクォークのカラー添字は明記せず、一般的なクォークのカラー三重項を単に q と表記する。

$SU(3)_c$ のゲージ場を G_μ とすると、これは一般的な $SU(3)$ 群の 8 つの生成子 $\widetilde{T}^\alpha=\frac{\lambda^\alpha}{2}$ $(\alpha=1,...,8)$ を用いて、

$$
G_\mu=G_\mu^\alpha\widetilde{T}^\alpha \tag{A.34}
$$

というように展開できる。ただし、λ^α は A.1 節に記したゲルマン行列である。よって、強い相互作用の寄与のみを考慮した共変微分 D_μ^c は

$$
D_\mu^c=\partial_\mu+ig_3G_\mu^\alpha\frac{\lambda^\alpha}{2} \tag{A.35}
$$

となる。ただし g_3 は $SU(3)_c$ ゲージ群の結合定数である。このゲージ場 $G_\mu^\alpha(x)$ がカラー荷を持つ粒子と相互作用するゲージボソンとして知られるグルーオン (gluon) である。このグルーオンの運動項は、

$$
-\frac{1}{4}G_{\mu\nu}^\alpha G^{\alpha\mu\nu} \tag{A.36}
$$

と書ける。ただし、

$$G^{\alpha}_{\mu\nu} = \partial_\mu G^{\alpha}_{\nu} - \partial_\nu G^{\alpha}_{\mu} - g_3 f^{\alpha\beta\gamma} G^{\beta}_{\mu} G^{\gamma}_{\nu} \tag{A.37}$$

であり、構造定数 $f^{\alpha\beta\gamma}$ は $SU(3)$ 群の生成子の交換関係から、

$$\left[\frac{\lambda^{\alpha}}{2}, \frac{\lambda^{\beta}}{2}\right] = i f^{\alpha\beta\gamma} \frac{\lambda^{\gamma}}{2} \tag{A.38}$$

として定義されたものである。式 (A.36) および QCD におけるクォークの運動項は、$SU(3)_c$ ゲージ変換

$$q \to q^{'} = e^{-ig_3 \widetilde{\Lambda}^{\alpha}(x) \frac{\lambda^{\alpha}}{2}} q \tag{A.39}$$

の下で不変になることが確認できる。ただし、$\widetilde{\Lambda}^{\alpha}(x)$ は $SU(3)_c$ ゲージ変換の局所的な位相である。

付録 B

ATLAS 検出器の補足

ATLAS 検出器の検出器群は主に内部飛跡検出器、カロリメータ、ミューオンスペクトロメータの 3 つに分類される。それぞれにはさらに複数種類のサブ検出器が含まれるが、本論では TGC を除くサブ検出器の説明は省略した。ここではそれらのサブ検出器について補う。

B.1 内部飛跡検出器

内部飛跡検出器は Pixel、Semiconductor Tracker (SCT)、および Transition Radiation Tracker (TRT) の 3 つのサブ検出器から構成される。

Pixel

Pixel は 4 つのシリコンピクセルのセンサーの層からできた検出器である。$R = 33.25$ mm のバレル部のみに位置する最内層は Insertable B-Layer (IBL) と呼ばれ、Run 2 から新たに導入された。そのシリコンピクセルの大きさは ϕ 方向に 50 μm、z 方向に 250 μm であり、その位置分解能は $(\sigma_\phi, \sigma_z) = (8\ \mu\mathrm{m}, 40\ \mu\mathrm{m})$ である。Run 1 からあった外側 3 層はそれぞれバレル部では layer-0、layer-1、layer-2、エンドキャップ部では disk と呼ばれる。これらの 3 層に使われているシリコンピクセルはすべて同じものである。それらの大きさは ϕ 方向に 50 μm で、z 方向に 400 μm であり、その位置分解能は $(\sigma_\phi, \sigma_{z(R)}) = (10\ \mu\mathrm{m}, 115\ \mu\mathrm{m})$ である。

Semiconductor Tracker (SCT)

Semiconductor Tracker (SCT) はシリコンストリップセンサーからなる検出器である。バレル部は 4 層の layer から、エンドキャップ部は片サイド 9 層の disk から構成されている。バレルの各層はビーム軸と平行にストリップが並んでいる片面のセンサー層に、40 mrad の角度だけずらして同様のセンサー層が背面同士でくっついた構造になっている。またエンドキャップの各層についても、ストリップが動径方向と平行なセンサー層に、40 mrad の角度だけずらしたセンサー層がくっついている。このようにして SCT では 2 重のストリップセンサー層を用いてステレオ 2 次元測定が実現されている。

バレル部にはただ 1 種類の長方形のストリップセンサーが採用されており、これらのストリップの間隔は 80 μm である。エンドキャップ部に関しては 5 種類の台形のストリップセンサーが採用されている。それらのストリップの間隔はおよそ 57–94 μm で、センサーの種類や各センサー上の位置によって異なる。ステレオ 2 次元測定を用いたときの SCT の位置分解能は $(\sigma_\phi, \sigma_{z(R)}) = (17\ \mu\mathrm{m}, 580\ \mu\mathrm{m})$ である。

Transition Radiation Tracker (TRT)

Transition Radiation Tracker (TRT) は直径 4 mm のストロー型のドリフトチューブを用いた検出器である。バレル部のチューブはビーム軸と平行な方向に積み重ねられており、その長さは 144 cm である。他方でエンドキャップ部のチューブは R 方向に伸びるように設置されており、その長さは 37 cm である。TRT のドリフトチューブはバレル部とエンドキャップ部の狭間を除く、$|\eta| < 0.8$ および $1.0 < |\eta| < 2.0$ の領域の粒子が少なくとも 36 本のドリフトチューブを通過するように配置されており、飛跡の再構成を精密に行えるような設計になっている。なお Pixel と SCT は $|\eta| < 2.5$ の領域を覆っているが、TRT は $|\eta| < 2.0$ までの領域しか覆っていない。TRT の位置分解能は 130 μm である。

TRT はその名の通り、遷移輻射を活用することで電子の識別を可能にしている。バレルおよびエンドキャップのチューブの間にはそれぞれカーボンファイバーおよびポリプロピレンのフォイルが存在し、荷電粒子がこれらを通過すると遷移輻射による光子が放出される。この光子はドリフトチューブ内のキセノンガスによって吸収され、それに続くイオン化によって、チューブの中心を通るアノードワイヤーにより多くの電子が集まることで遷移輻射を検知できる。加えて ATLAS 検出器内では、エネルギーが 100 GeV 以下の粒子に限定した場合、電子しか遷移輻射を起こすような速度に達しないため、遷移輻射の有無によって電子を識別することができる。このようにして TRT は飛跡の再構成に加え、電子の識別という重要な役割を担っている。

B.2 カロリメータ

ATLAS 検出器のカロリメータの断面図は図 3.8 に示した通りである。ATLAS 検出器のカロリメータは用途別に、電磁カロリメータとハドロンカロリメータの 2 つに大きく分けられる。さらにこれらは液体アルゴンおよび吸収体として鉛を用いた Liquid Argon (LAr) カロリメータと、シンチレータおよび吸収体として鉄を用いた Tile Calorimeter から構成されている。

電磁カロリメータ

電磁カロリメータは電子と光子の位置およびエネルギーを測定する。図 3.8 の LAr electromagnetic barrel と LAr electromagnetic end-cap (EMEC) がこれに対応する。LAr カロリメータの動作温度は約 88.5 K なので、これらはすべてクライオスタットに内蔵されている。電磁カロリメータの最大の特徴は、鉛の層と液体アルゴンの層からなるアコーディオン型構造である。図 B.1 にバレル電磁カロリメータの構造を示す。このようなアコーディオン型構造を採用することで、隙間を作らずに ϕ 全体を覆うことができ、安定したパフォーマンスを期待することができる。

バレル部は $|\eta| < 1.475$ をカバーし、バレルクライオスタットに内蔵されている。電磁放射長 (radiation length) X_0 に対してバレル部は少なくとも $22X_0$ の厚みを持っており、電磁シャワーが十分に吸収されることが保証されている。図 B.1 からもわかるように、バレル電磁カロリメータは layer 1、2、3 の 3 つの層から構成されている。Layer 1 は R 方向の厚みが $4.3X_0$ の薄いストリップセルから構成されており、その $|\eta| < 1.4$ における granularity (カロリメータセルの大きさ) は $\Delta\eta \times \Delta\phi = 0.0031 \times 0.1$ である。Layer 2 は $16X_0$ と最も厚く、ほとんどのエネルギーがこの層で落とされる。Layer 2 の $|\eta| < 1.4$ における granularity は $\Delta\eta \times \Delta\phi = 0.025 \times 0.025$ である。Layer 3 は厚みが $2X_0$ で、granularity は $\Delta\eta \times \Delta\phi = 0.050 \times 0.025$ である。また layer 1 より内側には presampler という、カロリメータに到達する前のエネルギー損失を補正するための薄い液体アルゴン層を持つ検出器が設置されている。Presampler の granularity は $\Delta\eta \times \Delta\phi = 0.025 \times 0.1$ となっている。表 B.1 にバレルおよび後述するエンドキャップの presampler と電磁カロリメータの各層の granularity と η の関係を示す。

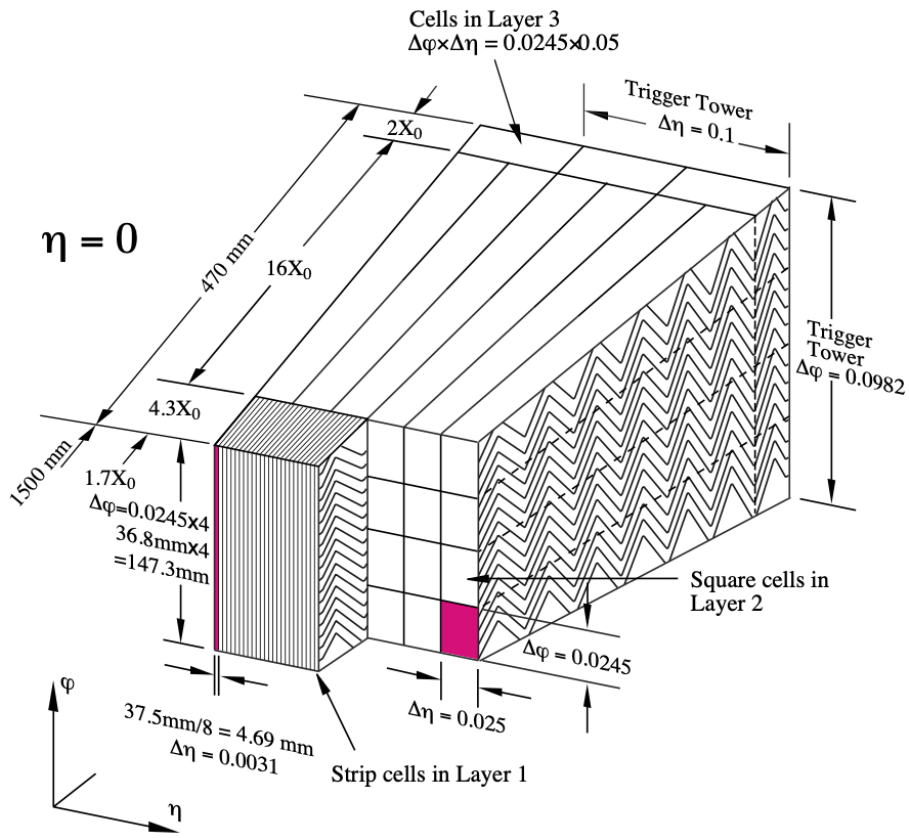


図 B.1　バレル電磁カロリメータの構造 [11]。波線が鉛の層と液体アルゴンの層のアコーディオン型構造を示している。エンドキャップも同じ構造を採用している。

表 B.1　電磁カロリメータの各 $|\eta|$ における granularity $\Delta\eta \times \Delta\phi$ [11]。

	Barrel		Endcap	
Presampler	$\lvert\eta\rvert < 1.52$	0.025×0.1	$1.5 < \lvert\eta\rvert < 1.8$	0.025×0.1
Layer 1	$\lvert\eta\rvert < 1.4$	$0.025/8 \times 0.1$	$1.375 < \lvert\eta\rvert < 1.425$	0.050×0.1
	$1.4 < \lvert\eta\rvert < 1.475$	0.025×0.025	$1.425 < \lvert\eta\rvert < 1.5$	0.025×0.1
			$1.5 < \lvert\eta\rvert < 1.8$	$0.025/8 \times 0.1$
			$1.8 < \lvert\eta\rvert < 2.0$	$0.025/6 \times 0.1$
			$2.0 < \lvert\eta\rvert < 2.4$	$0.025/4 \times 0.1$
			$2.4 < \lvert\eta\rvert < 2.5$	0.025×0.1
			$2.5 < \lvert\eta\rvert < 3.2$	0.1×0.1
Layer 2	$\lvert\eta\rvert < 1.4$	0.025×0.025	$1.375 < \lvert\eta\rvert < 1.425$	0.050×0.025
	$1.4 < \lvert\eta\rvert < 1.475$	0.075×0.025	$1.425 < \lvert\eta\rvert < 2.5$	0.025×0.025
			$2.5 < \lvert\eta\rvert < 3.2$	0.1×0.1
Layer 3	$\lvert\eta\rvert < 1.35$	0.050×0.025	$1.5 < \lvert\eta\rvert < 2.5$	0.050×0.025

片サイドに 1 つある EMEC は $1.375 < |\eta| < 3.2$ をカバーし、それぞれ後述の HEC や FCal とともにエンドキャップクライオスタットに内蔵されている。EMEC は inner wheel と outer wheel という 2 つの同軸の wheel から構成されている。$1.475 < |\eta| < 2.5$ の outer wheel 領域では少なくとも $24X_0$ の厚みを持ち、$2.5 < |\eta| < 3.2$ の inner wheel 領域では少なくとも $26X_0$ の厚みを持つ。バレルと同様に、EMEC も 3 つの層から構成されており、layer 1 の手前には presampler がある。EMEC の granularity は $|\eta|$ に依って頻繁に変わるので、詳しくは表 B.1 を参照していただきたい。

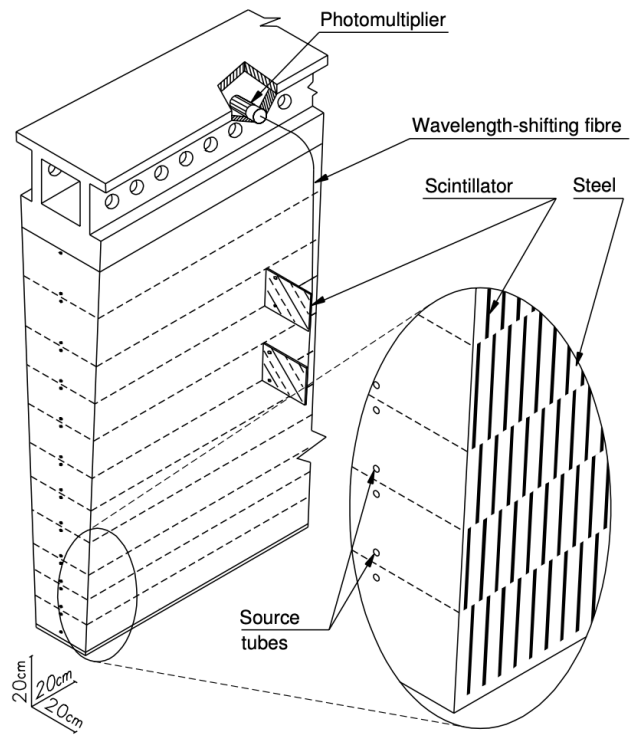


図 B.2　Tile Calorimeter のモジュールの構造 [11]。Barrel および extended barrel は 64 個の $\Delta\phi = 5.625°$ のモジュールに分割されている。右に拡大されているように、シンチレータタイルと鉄のサンドイッチ構造になっている。

ハドロンカロリメータ

ハドロンカロリメータはジェットなどのハドロンのエネルギーを測定する。ハドロンカロリメータは Tile Calorimeter、LAr Hadronic End-cap (HEC)、および LAr Forward (FCal) の 3 種類のカロリメータから構成される。

Tile Calorimeter は前述したように、シンチレータタイルと吸収体の鉄がサンドイッチ構造になったカロリメータで、$|\eta| < 1.0$ をカバーする barrel 領域と $0.8 < |\eta| < 1.7$ をカバーする extended barrel 領域に分けられている。どちらにおいても内径は 2.28 m、外径は 4.25 m であり、これはハドロンの相互作用長 (interaction length) λ_0 を用いるとおよそ $7.4\lambda_0$ に相当する。Barrel と extended barrel はともに図 B.2 に示すように ϕ 方向に 64 個のモジュールという部分に分割されている。各モジュールにおける シンチレータの信号は光電子増倍管によって読み出されるが、その読み出しは R 方向に分割された層の単位で行う。本論の図 3.9 にその層の配置は示した。$\eta = 0$ では内側から、厚さが $1.5\lambda_0$ の A 層、厚さが $4.1\lambda_0$ の B/C 層、そして厚さが $1.8\lambda_0$ の D 層に分かれている。特に extended barrel の D 層は Run 2 の途中からミューオントリガーにも活用されるようになった。

HEC は EMEC の外側にある、ハドロン用の LAr カロリメータである。鉛が吸収体の電磁カロリメータとは異なって、銅を吸収体として採用している。HEC は EMEC と同じエンドキャップクライオスタットの中に入っており、 $1.5 < |\eta| < 3.2$ の領域を覆う。検出器自体は z 方向に 2 つの wheel という単位に分かれており、さらに各 wheel は z 方向に 2 つの section に分かれている。Granularity は $1.5 < |\eta| < 2.5$ の領域で $\Delta\eta \times \Delta\phi = 0.1 \times 0.1$ で、$2.5 < |\eta| < 3.2$ で $\Delta\eta \times \Delta\phi = 0.2 \times 0.2$ である。

FCal はエンドキャップ領域よりさらに $|\eta|$ が大きいフォワード領域を覆う、電磁およびハドロン用の LAr カロリメータである。$3.1 < |\eta| < 4.9$ の領域をカバーし、EMEC や HEC と同じエンドキャップクライオスタットに内蔵されている。FCal は z 方向に 3 つの検出器単位に分かれているが、1 層目は電磁カロリメータとして、2 層目と 3 層目はハドロンカロリメータとして使われている。吸収体に関しては 1 層目には銅を、2 層目と 3 層目にはタングステンを用いている。図 B.3 にハドロンカロリメータ層の interaction length を示す。

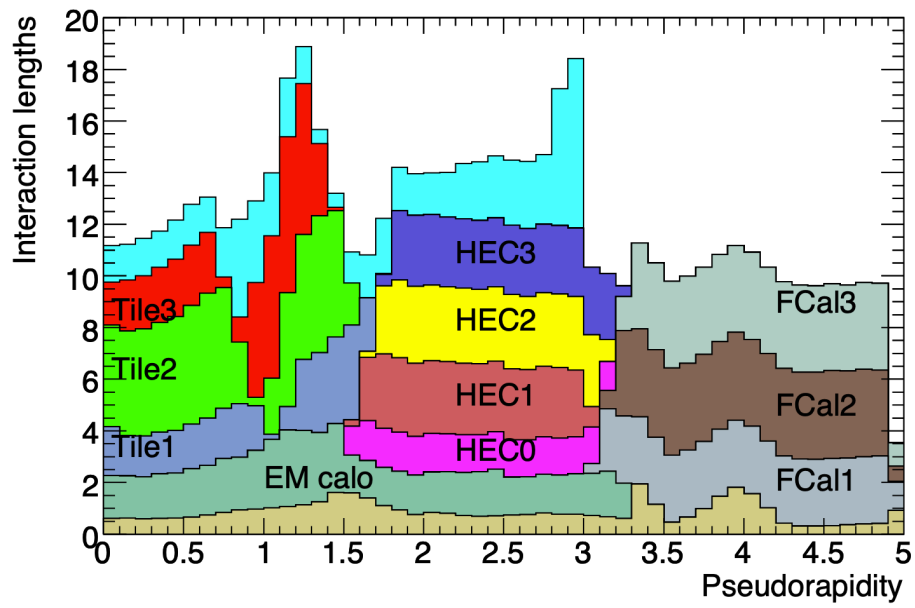


図 B.3　ハドロンカロリメータ層の interaction length [11]。横軸が $|\eta|$、縦軸がその $|\eta|$ における各カロリメータの層の interaction length になっている。HEC 0–3 は 2 つの HEC wheel のそれぞれにある 2 つの section に相当する。$|\eta| < 3.0$ のみに描かれている水色の領域はミューオンスペクトロメータとカロリメータ層の間にある物質の interaction length を表している。

B.3　TGC 以外のミューオンスペクトロメータ

ミューオンスペクトロメータは、ミューオンの運動量の精密測定を行う MDT と CSC、およびミューオンの運動量を概算してトリガーを発行を行う RPC と TGC という 4 種類のサブ検出器から構成される。本研究と密接に関係する TGC 検出器についてはすでに本論の 3.2.5 節で述べた。ここでは本論で説明しなかった MDT、CSC、および RPC の詳細について述べる。

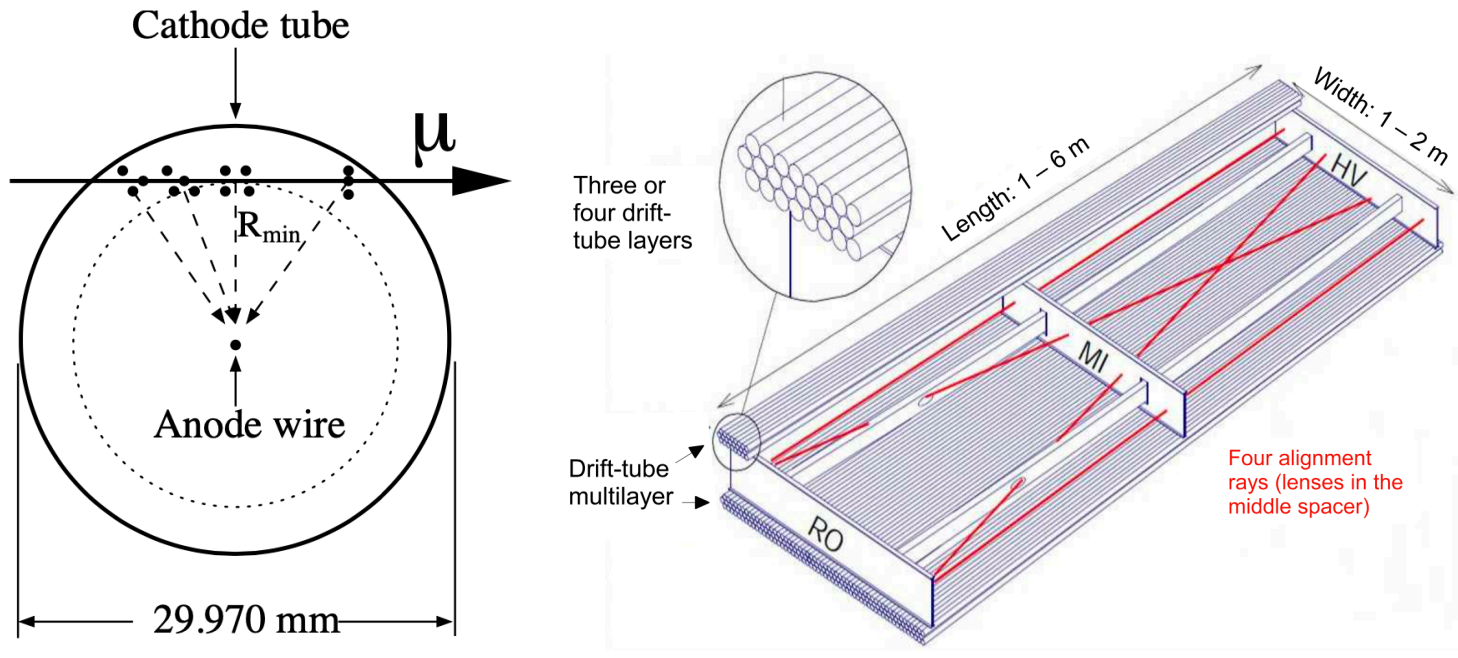
Monitored Drift Tube (MDT)

Monitored Drift Tube (MDT) はミューオンの運動量の精密測定を行う検出器であり、バレルおよびエンドキャップ領域ともに 3 層のチェンバーから構成されている。そのカバー領域は最内層を除く 2 層で $|\eta| < 2.7$ であり、最内層で $|\eta| < 2.0$ である。図 3.10 や図 3.11 にも記されているように、各チェンバーには英字 3 文字の名前がついている。最初の文字は barrel か endcap かを区別するために B または E がついている。2 文字目は inner、middle、outer の各層に対応する I、M、L のいずれかが原則ついている[*1]。3 文字目はチェンバーが位置するセクターを特定するための文字で、原則 small の S または large の L がついている[*2]。

ドリフトチューブは直径 29.970 mm のアルミニウム製で、各チューブの中にはタングステン–レニウム合金からできた直径 50 μm のアノードワイヤーが通っている。図 B.4(a) にその断面図を示す。チューブ内は混合比 93:7 の Ar/CO_2 ガスで満たされており、荷電粒子が通過するとガスが電離し、生じた電子は 3080 V の印加電圧によって増幅され、アノードワイヤーによって回収される。この際の電子のドリフト時間を測定することで、ミューオンの通過位置を決定する。ドリフト時間は最大でおよそ 700 ns で、チューブあたりの位置分解能は 80 μm であると見積もられている。

[*1] バレルとエンドキャップの間の transition 領域に位置するチェンバーはこの例外である。BEE は Barrel Endcap Extra の略で、EES/EEL は Endcap Extra Small/Large の略である。

[*2] ATLAS 検出器を支えるための支柱があるセクター 12 や 14 のチェンバーはこの原則に従わない (図 3.10(b) 参照)。これらのセクターでは支柱によるアクセプタンスの損失を最小限に保つために、BIR や BOF/G などの特別なチェンバーを採用している。



(a) ドリフトチューブの構造　　(b) MDT チェンバーの構造

図 B.4　MDT のチューブとチェンバーの構造 [11]。図 B.4(a) はドリフトチューブの構造。ミューオンが通過することで生じる電子のドリフト時間を測定することで、ミューオンの通過位置を決定する。図 B.4(b) はチェンバーの構造。この図のチェンバーは 3 つのドリフトチューブ層からなる multilayer が 2 つある、2×3 のチェンバーである。赤い線はアライメントシステムに使われている光線を表している。

MDT のチェンバーの模式図を図 B.4(b) に示す。各チェンバーは必ず multilayer という 3 層あるいは 4 層のドリフトチューブの層から構成されている。ほとんどのチェンバーは図 B.4(b) のように 2 つの multilayer を含む[*3]。チェンバーの枠組はアルミニウム製で強固な構造になっているが、経年劣化などによる歪みを検知できるように、光線を用いたアライメントシステムが組み込まれている。このシステムはチューブと平行な 2 本の光線とチェンバーを斜めに横切る 2 本の光線を使って、数 μm の歪みを検知することができる。

Cathode Strip Chamber (CSC)

Cathode Strip Chamber (CSC) は $2.0 < |\eta| < 2.7$ のエンドキャップ領域に位置する、ミューオンの運動量を精密測定するための検出器である。図 3.11 からもわかるように、MDT の EIS/EIL チェンバーと並んで設置されている。この領域では粒子の通過頻度が非常に高く、MDT が安定動作するための閾値である 150 Hz/cm^2 を超えてしまうため、それに耐えられる CSC が代わりに用意されている。

図 B.5 に CSC 検出器の全体図を示す。CSC は small sector に属する 8 つの CSCS と large sector に属する 8 つの CSCL という 2 種類計 16 個のチェンバーから構成される。 各チェンバーには動径方向にアノードワイヤーが 2.5 mm 間隔で通っており、動径方向およびそれと垂直な ϕ 方向にカソードストリップが張り巡らされている。図 B.6 にそれらの概要を示す。ミューオンが CSC チェンバーを通過すると混合比 80:20 の Ar/CO_2 ガスが電離し、1900 V の印加電圧によって電子雪崩が起こる。この際のストリップ上の電荷の分布を読み出すことで、ミューオンの通過位置を決定している。その位置分解能は CSC の平面に沿って 60 μm で、検出器の時間分解能も 7 ns と良く、粒子の高い通過頻度に対応できるような設計となっている。

[*3] 例外として、BEE と BIS8 は 1 つの multilayer から構成される。

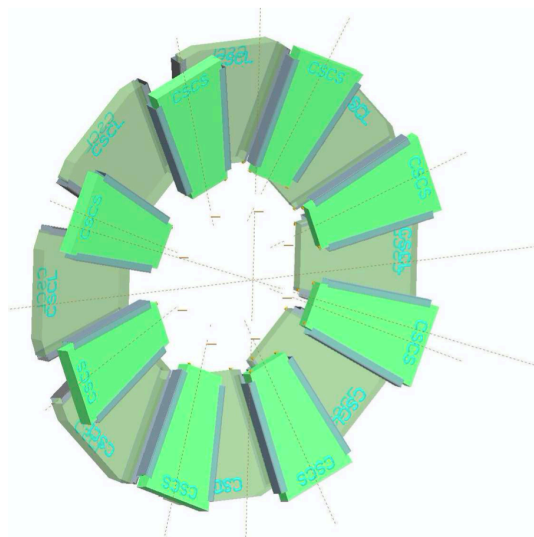
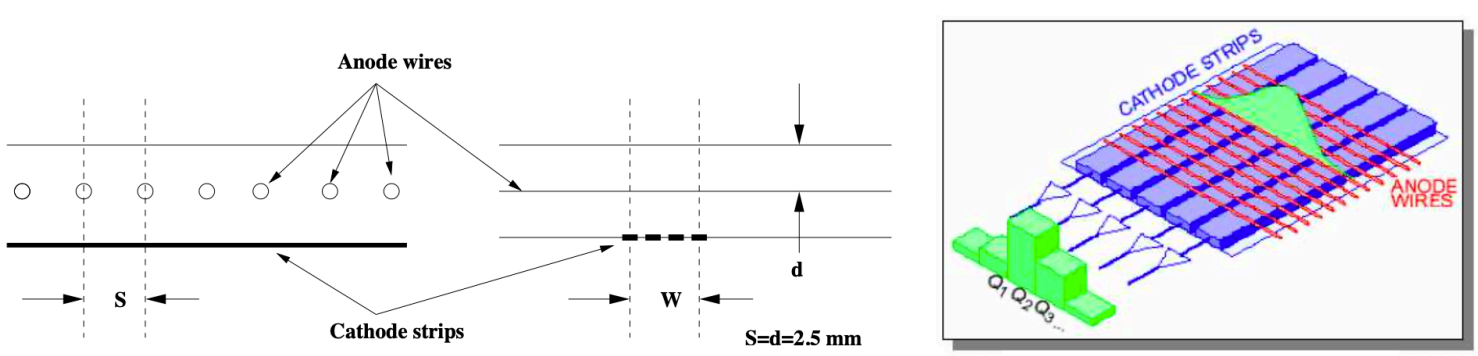


図 B.5　CSC の全体図 [11]。Small sector (CSCS) と large sector (CSCL) の 2 つのチェンバーによって構成される。



(a) アノードワイヤーとカソードストリップの配置　　(b) カソードストリップの読み出し

図 B.6　CSC のアノードワイヤーとカソードストリップ [11]。図 B.6(a) はアノードワイヤーとカソードストリップの配置。アノードワイヤー同士の間隔 S およびアノードとカソードの間隔 d はともに 2.5 mm である。W はストリップの読み出し間隔を表しており、CSCS と CSCL のそれぞれで $W = 5.308, 5.567$ mm である。図 B.6(b) はカソードストリップの読み出しを表している。アノードとの電圧によって生じるカソード上の電荷分布を読み出すことでミューオンが通過した位置を決定する。アノードワイヤーの信号は読み出さない。

Resistive Plate Chamber (RPC)

Resistive Plate Chamber (RPC) は $|\eta| < 1.05$ のバレル領域に位置するミューオントリガー検出器である。図 3.11 に示されているように、RPC は 3 つのステーションと呼ばれる層から構成されており、それぞれのステーションは衝突点に近い側から順に RPC1、RPC2、RPC3 と呼ばれる。RPC1 と RPC2 は MDT の BMS/BML チェンバーを挟む形で配置されており、RPC3 は BOS/BOL の衝突点に近い側に位置する。RPC1 と RPC2 は 4–10 GeV の p_{T} 閾値を用いて低運動量のミューオンを捉えるために使われ、RPC3 は 11–20 GeV の p_{T} 閾値を用いてより高い運動量を持つミューオンを捉えるために使われている [45]。各ステーションは (η, ϕ) の 2 次元測定ができる層を 2 つ含むので、すべてのステーションを通過したミューオンは 6 層に 2 次元ヒット情報を残すことになる。

RPC のチェンバーの断面図を図 B.7 に示す。RPC のチェンバーはメラミンフェノール樹脂からできた、厚さ 2 mm の 2 枚の高抵抗のプラスチック板の間に、2 mm のガス層が挟まれているという基本構造を持つ。使用しているガスは $C_2H_2F_4(94.7\%)/\mathrm{Iso}-C_4H_{10}(5.0\%)/SF_6(0.3\%)$ の混合ガスである。ガス層における電場は 4.9 kV/mm で、荷電粒子が通過すると電子雪崩が起こる。この信号は ϕ 方向と z 方向のそれぞれに伸びる 2 次元ストリップによって読み出され、通過したミューオンの (η, ϕ) が決められる。各ストリップの幅は 23–35 mm で、チェンバーの

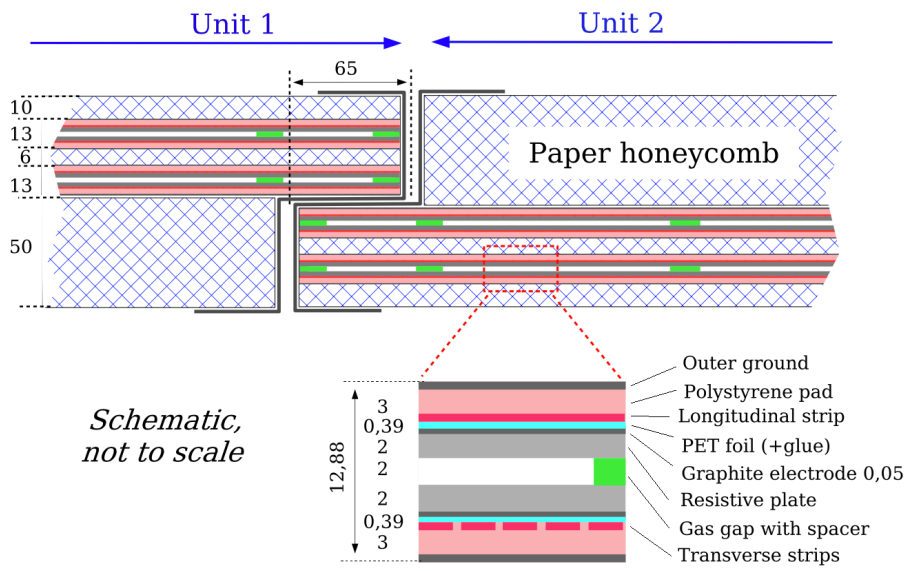


図 B.7　RPC のチェンバーの断面図 [11]。長さの単位はすべて mm である。各チェンバーは 2 つの unit という部分から構成されており、この図は 2 つの unit の境界付近の模式図である。各 unit には 2 つのガス層があり、それぞれで (η, ϕ) の測定を行う。ミューオンが通過した際の信号はガス層と高抵抗板を挟んで配置されている 2 次元ストリップによって読み出される。η 方向の測定を行う η-ストリップは紙面と垂直な方向に伸び、ϕ 方向の測定を行う ϕ-ストリップは紙面と平行な方向に伸びている。

位置分解能は z 方向と ϕ 方向ともに 10 mm となっている。検出器のみの時間分解能は 1.5 ns である。

付録 C

OKS データベースの詳細

本論では OKS データベースの説明は概要のみにとどめ、その詳細は省略した。ここでは OKS のオブジェクトについて、OKS の制御について、および本研究で開発した Level-1 ミューオンエンドキャップトリガーと TGC の OKS の詳細について記す。

C.1 OKS のオブジェクト

OKS データベース上のファイルが schema ファイルと data ファイルの 2 種類から構成されることは 5.1.1 節でも述べたとおりである。ここではそれらを用いてどのように OKS のオブジェクトが定義されるか、および検出器システムの階層構造を定義する上で重要なオブジェクトの具体的な定義について説明する。

図 5.2 にも示したように、schema ファイルで定義する class には attribute、relationship、superclass などの複数の “リスト” を含ませることができる[*1]。Attribute とはその class に属する単一パラメータのことであり、type によってその型を指定できる。Relationship はその class をまた別の class と紐付けたい場合に用いる。Superclass は別の class の attribute や relationship をまとめて引き継ぎたい場合に用いる。Class A の attribute や relationship をまとめて class B で利用した場合は、class B の superclass として class A を定義すれば良い。

Data ファイルでは、schema ファイルで定義した class を用いて固有の id を持つオブジェクトを定義する。この id は一意でなければならず、場合によっては class が異なっていても同じ OKS データベース内で同じ id を持つオブジェクトを定義することは許されない。オブジェクトの attribute や relationship の設定も data ファイル内で行う。

OKS の中心的な管理は ATLAS の TDAQ グループが行なっている。各サブシステムは TDAQ グループによって用意された class を用いて、自らのシステムに適合した階層構造を定義しなければならない。TDAQ グループによって用意されている数ある class のうち、システムの階層構造を定義する上で重要な class の概要を以下にまとめる。

Partition

Partition は検出器システムを統合運転するための最小単位であり、その統合運転に必要なシステムをすべて含む最上位の class である。Partition の代表的な relationship には Segments、Disabled、MasterTrigger などがある。Segments は Partition に後述の Segment を紐付けるためのもので、その Partition が含むべき検出器システムを指定するために用いられる。Disabled とはその Partition に含まれるシステムの中で使わない検出器などを指定するために用いる。検出器の不具合などによって一時的に検出器の一部を使用したくない場合などは、対応するオブジェクトを Partition から引き抜くのではなく、Disabled のリストに入れる。MasterTrigger とはその Partition を走らせる際に、どのモジュールを用いてトリガー信号を発行するかを指定するための relationship である。

[*1] Method というリストも存在するが、ほとんど使われていないのでここでは説明を省く。

Segment

Segment は独立したシステム単位を定義するための class である。ここで言う独立したシステムとは、他の TDAQ システムには依存せずに、それ単体で制御が可能であるようなシステムを指す。Segment の代表的な relationship には Segments、Resources、IsControlledBy などがある。Segments relationship は Partition の Segments relationship と全く同じで、配下に置きたい Segment を指定するために用いられる。Resources relationship は後述の Resource を Segment の配下に置くために用いられる。IsControlledBy とは、その Segment を制御するために用いるアプリケーションを指定するための relationship である。Segment オブジェクトを用意する際は、対応する PC や SBC で走る Run Control Application も用意しなければならないことは 5.1.1 節でも述べた。

Resource

Resource は Partition の Disabled relationship に加えることで disable することができる、検出器システムの階層構造における最小単位である。多くの場合は 1 台のエレクトロニクスやアプリケーションが 1 つの Resource オブジェクトに対応する。例えばハードウェアの不具合によって特定のボードが使えなくなった場合、そのボードを単体で Partition から disable することができる。

ResourceSet

ResourceSet とは、Resource の集合体を作るために用意されている class である。複数の関連した Resource を ResourceSet にまとめることで、Partition からの disable の操作が効率的にできるようになる。ある ResourceSet を Partition から disable すると、その ResourceSet に含まれる Resource はすべて disable される。ResourceSet オブジェクトに Resource を含めたい場合は、その ResourceSet の Contains という relationship に該当する Resource オブジェクトを列挙すれば良い。

ResourceSetAND

ResourceSetAND は superclass として ResourceSet を持つ class で、配下にある Resource オブジェクトがすべてPartition から disable された場合、自身も自動で disable される。

ResourceSetOR

ResourceSetOR は superclass として ResourceSet を持つ class で、配下にある Resource オブジェクトが 1 つでもPartition から disable された場合、自身も自動で disable される。

5.1.1 節で述べたように、検出器システムを正しく制御するためには、これらを駆使して適切なシステムの階層構造 (Segments & Resources) を開発する必要がある。

C.2 OKS データベースの制御

ここでは OKS データベースがどのようにして読み込まれるかについて補足しておく。

OKS の制御は ATLAS の TDAQ グループが開発している、C++ で書かれた OKS 制御 API によって行われる。検出器の運転を開始するために Partition を起動すると、まずは使おうとしている OKS データベースの整合性が OKS 制御 API によって確認される。オブジェクトの id に重複があるなど、OKS に何かしらの問題がある場合はこの段階で Partition の起動に失敗する。整合性が取れている場合は、必要な schema ファイルと data ファイルが読み込まれ、図 5.1 に示したように RootController が起動する。RootController が起動した後に OKS を変更した場合は再度データベースを読み込む必要があるが、所要時間は数秒程度である。

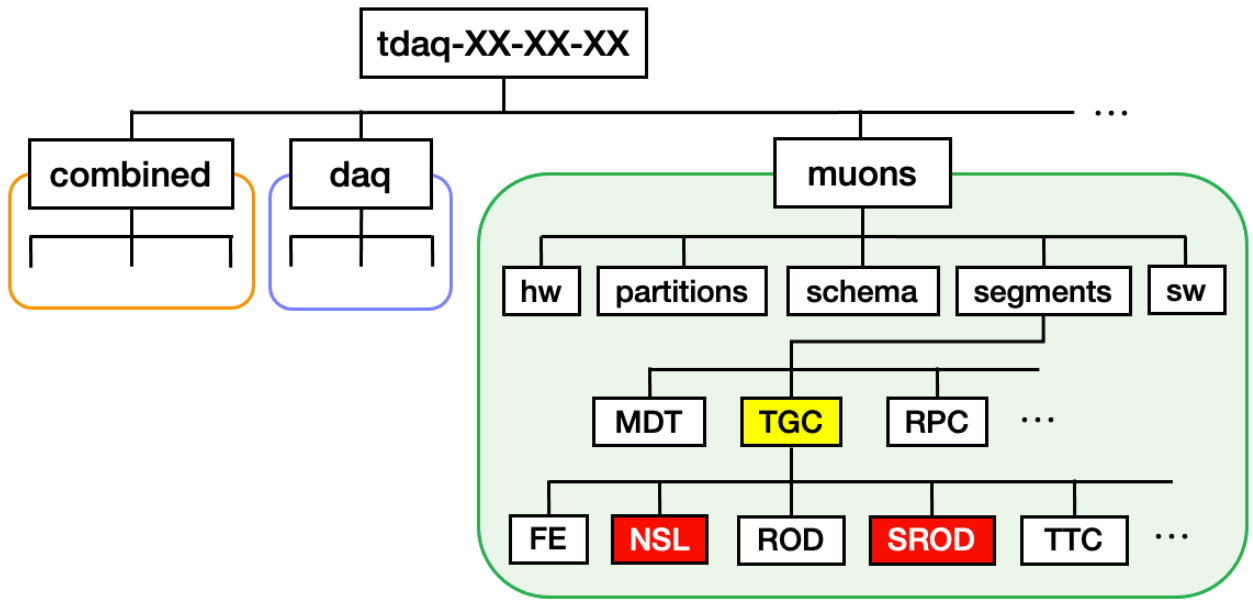


図 C.1　OKS データベースのファイル構造。TGC などのミューオン検出器の情報はすべて muons ディレクトリに入っている。combined ディレクトリには物理データ取得の際に使う ATLAS Partition などの複数の検出器を走らせるためのデータが入っている。daq ディレクトリには全サブシステムで共通して使う class を定義した schema ファイルなどが入っている。

C.3　Level-1 ミューオンエンドキャップトリガーおよび TGC の OKS の詳細

本節では本研究で開発した L1MuE トリガーおよび TGC の統合制御を実現するために開発した OKS データベースの詳細について記述する。

C.3.1　OKS データベースのファイル構造

OKS データベースは前述したように多数の XML 形式のファイルによって構成される。それらのファイル構造を図 C.1 に示す。OKS データベースはすべて tdaq-XX-XX-XX という名前のディレクトリに入っており、“XX-XX-XX” の部分は TDAQ ソフトウェアのバージョンを表す[*2]。その下には各サブシステムに対応するディレクトリがあり、ミューオン検出器のデータはすべて muons ディレクトリにまとめられている。muons ディレクトリは、hw、partitions、schema、segments、sw という 5 つのサブディレクトリから構成される。hw には使用する PC や SBC の情報が含まれる。partitions にはミューオン検出器のみを用いる、Standalone Partition を定義したファイルが入っている。schema ディレクトリには各ミューオン検出器のデータベースを定義するための schema ファイルが、sw には参照するオンラインソフトウェアパッケージを指定するためのファイルが含まれる。segments ディレクトリには図 5.8 に示したような Segments & Resources を定義するためのファイルが含まれる。

L1MuE トリガーおよび TGC のコンフィギュレーションパラメータや Segments & Resources は segments ディレクトリの配下にある TGC ディレクトリの下にあるファイル群で定義されている。TGC ディレクトリの配下にある FE は TGC のフロントエンドエレクトロニクスを、ROD は TGC ROD を、そして TTC は TGC の TTC システムを制御するために用意されている。これらと並ぶ形で、New SL クレートを制御するための NSL ディレクトリ、SROD を制御するための SROD ディレクトリを新たに設けた。

[*2] 2021 年 1 月現在の TDAQ ソフトウェアのバージョンは 09-02-01 である。

C.3.2 New SL クレートを制御するために開発した OKS の詳細

Run 3 における L1MuE トリガーの統合制御を実現するべく、新たに導入される New SL クレートを制御するための OKS データベースを開発した。具体的には C.3.1 節で触れた NSL ディレクトリの開発に加え、各エレクトロニクスの class を定義した schema ファイルや、L1MuE トリガー用のソフトウェアレポジトリを新たに導入した。本小節では新たに導入したデータベース群について説明する。

5.1.1 節で述べたように、各エレクトロニクスのコンフィギュレーションパラメータを設定するためには、そのエレクトロニクスに対応する OKS class とオブジェクトを定義する必要がある。そこでまず、図 C.1 にも示した muons/schema ディレクトリの下に New SL、TTCFOB、G-Link Converter、Burst Stopper のための class を定義した schema ファイル (tgcNewSL.schema.xml, tgcTTCFanout.schema.xml, tgcCnv.schema.xml, tgcBs.schema.xml) を用意した。これらの schema ファイルにはそのエレクトロニクス自体を表す class、さらにはそのエレクトロニクスの class が relationship として持つ各コンフィギュレーションパラメータの class が定義してある。ここでは各エレクトロニクスの class のみを抜粋して、その定義を表 C.1–C.4 に載せる。なお表 C.1 は表 5.3 に “attribute/relationship” の欄を付け加えたものである。

表 C.1 New SL の OKS class (NewSLBoard) の定義。

名称	attribute/relationship	概要
BoardAddress	attribute	New SL ボードの 32 ビット VME アドレス。
BoardType	attribute	Endcap または forward。
BoardId	attribute	ボードを識別するための ID。E01–E24 あるいは F01–F12 のいずれか。
BitFile	relationship	FPGA に書き込むビットファイル。
LUTFile	relationship	FPGA に書き込む LUT ファイル。
DelayFPGA	relationship	FPGA で受信したヒット信号や TTC 信号にかける遅延時間。
MaskFPGA	relationship	FPGA で受信したヒット信号や TTC 信号にかけるマスクパラメータ。
InnerCoincidenceFlagFPGA	relationship	FPGA のトリガーパスにおける Inner Coincidence にどの検出器を用いるかを表すフラグ。
ReadoutFPGA	relationship	FPGA の読み出しパスのパラメータ。読み出す BC の数や Level-1 Buffer の大きさなどを含む。

表 C.2 TTCFOB の OKS class (TTCFanoutBoard) の定義。

名称	attribute/relationship	概要
BoardAddress	attribute	TTCFOB の 32 ビット VME アドレス。
BoardId	attribute	ボードを識別するための ID。CSL01–CSL06 のいずれか [*3]。
TTCrxI2cId	attribute	TTCrx の I^2C ID。
BCRDelay	attribute	受信した BCR 信号にかける遅延時間。
BitFile	relationship	FPGA に書き込むビットファイル。
BusyEnable	relationship	Busy 信号の発行元の enable と disable。
TTCrxRegisters	relationship	TTCrx の各レジスタの設定値。

表 C.3　G-Link Converter の OKS class (CnvBoard) の定義。

名称	attribute/relationship	概要
BoardAddress	attribute	Converter の 32 ビット VME アドレス。
BoardId	attribute	ボードを識別するための ID。CSL02、CSL03、CSL05、CSL06 のいずれか。
ConverterID	attribute	New SL に送信するデータに付与する整数型の ID。
BitFile	relationship	FPGA に書き込むビットファイル。
DelayFPGA	relationship	FPGA で受信したヒット信号や TTC 信号にかける遅延時間。
MaskFPGA	relationship	FPGA で受信したヒット信号や TTC 信号にかけるマスクパラメータ。

表 C.4　Burst Stopper の OKS class (BsBoard) の定義。

名称	attribute/relationship	概要
BoardAddress	attribute	Burst Stopper の 32 ビット VME アドレス。
FPGA	relationship	FPGA に書き込むビットファイル。
Delay	relationship	FPGA で受信したヒット信号にかける遅延時間。
Mask	relationship	FPGA で受信したヒット信号にかけるマスクパラメータ。
BurstStopper	relationship	バーストを判断する際に使用するロジックのパラメータ。
LUTThre	relationship	バーストを判断する際の信号数の閾値。
Output	relationship	出力信号のマスクパラメータなど。

次に、schema ファイルに定義した class を用いて実際のオブジェクトを定義した data ファイルを用意する必要がある。これらはすべて図 C.1 の segments ディレクトリの配下にある NSL ディレクトリの下に用意した。その一例はすでに本論の図 5.12 に示した。全 72 台の New SL に対応した形で 72 個の NewSLBoard オブジェクトを用意し、それらのオブジェクトが relationship として持つパラメータも New SL ごとに調整できるように、必要な数だけ定義した。さらに、TTCFOB、G-Link Converter、Burst Stopper についても同様の data ファイルを準備することで、全 6 個の New SL クレートに設置されているすべてのエレクトロニクスを制御するためのデータベースを構築した。

OKS データベースで設定した値を実際に Partition を走らせる際に使用するには、ソフトウェアアプリケーション側にも適切な実装が必要である。具体的には、ソフトウェアアプリケーションは各エレクトロニクスに対応するオブジェクトに含まれる attribute や relationship などの OKS の class の構造を事前に知っていなければならない。そのため、コンフィギュレーションパラメータに関係する class は、オンラインソフトウェアパッケージが持つ schema ファイルでも定義しなければならない。New SL クレートのエレクトロニクスの場合は、図 5.11 に示した L1MuESLModule、L1MuETTCFanoutModule、L1MuEGLinkCnvModule、L1MuEBurstStopperModule がそれに該当する。これらのパッケージに含まれる schema を OKS 側の schema と整合させることで[*4]、Partition を

[*3] CSL は Crate Sector Logic の略。

[*4] 典型的なミスとして、OKS の schema に必要な class を定義したが、ソフトウェアパッケージの schema にその class を定義し忘れた状況で、その class をソフトウェアアプリケーションで参照しようとするとパターンがある。この場合は OKS 自体の整合性は取れているので Partition は起動するが、ソフトウェアアプリケーションを走らせようとすると CONFIG から先に進めなくなる。

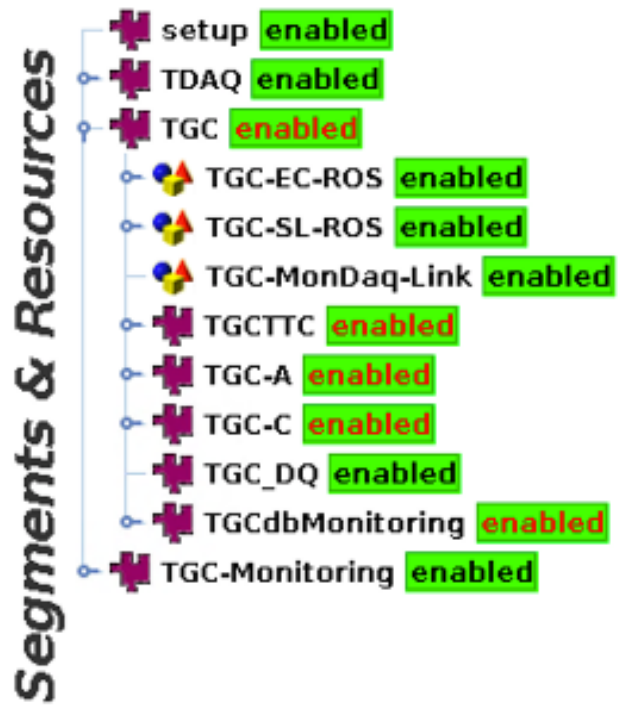


図 C.2　DAQSlice Partition の Segments & Resources の全体像。

走らせた際に OKS で定義した適切な値がコンフィギュレーションの際に書き込まれるようにした。

加えて、参照した OKS データベースを適切なオンラインソフトウェアパッケージと紐付ける必要がある。これは図 C.1 の muons ディレクトリの配下にある sw ディレクトリ内のファイルで行う。Run 3 から新たに導入する L1MuE ソフトウェアパッケージは Run 2 でも使用されていた TGC パッケージとは独立したものなので、編集のしやすさの向上やコンパイル時間の短縮のために、分離する (別ディレクトリで管理する) ことにした。このため、L1MuE パッケージを使用するためのソフトウェアレポジトリファイルを新たに用意した。このレポジトリファイルは第 6 章でも述べたように、SROD 関係のアプリケーションを定義するためにも使っている。

C.3.3　TGC の Segments & Resources の全体的な吟味と再編

5.3 節で説明したように、本研究では New SL クレートの新規エレクトロニクスを統合制御するための Segment や Resource を開発した。これらを Run 2 でも使用していた TGC システムと同期させるために、TGC の Segments & Resources の整合性を全体的に吟味し、再編した。本小節では新たに構築した TGC の Segments & Resources について詳細に述べる。

図 C.2 に主に開発を行う際に使用した TGC の Standalone Partition である DAQSlice (part_TGC_DAQSlice) が含む Segment の概要を示す。DAQSlice Partition には setup、TDAQ、TGC-Monitoring、TGC という 4 つの Segment が含まれる。setup は RootController によって制御される Segment で、Partition の全体的な設定を行うために用意されている。TDAQ Segment は図 3.17 に示したような HLT 関係の Resource などを含む Segment で、Standalone Partition で SFO にデータを流すために必要である。TGC-Monitoring という Segment は TGC 関係のオンラインヒストグラムを作成するためのものである。そして TGC は TGC および L1MuE トリガーに関係するすべてのエレクトロニクスの Segment や Resource を含む最上位の Segment であり、本研究ではこれを中心に開発した。

TGC Segment に含まれる Segment と Resource についても見ていく。TGC-EC-ROS および TGC-SL-ROS はそれぞれ ROD および SROD が出力したデータを処理する ROS と通信するための ResourceSet である。特に TGC-SL-ROS は第 6 章で説明した SROD の開発過程で新たに導入したものである。TGC-MonDaq-Link は TGC Segment の enable/disable を TGC-Monitoring Segment に伝播するために用意されている Resource であ

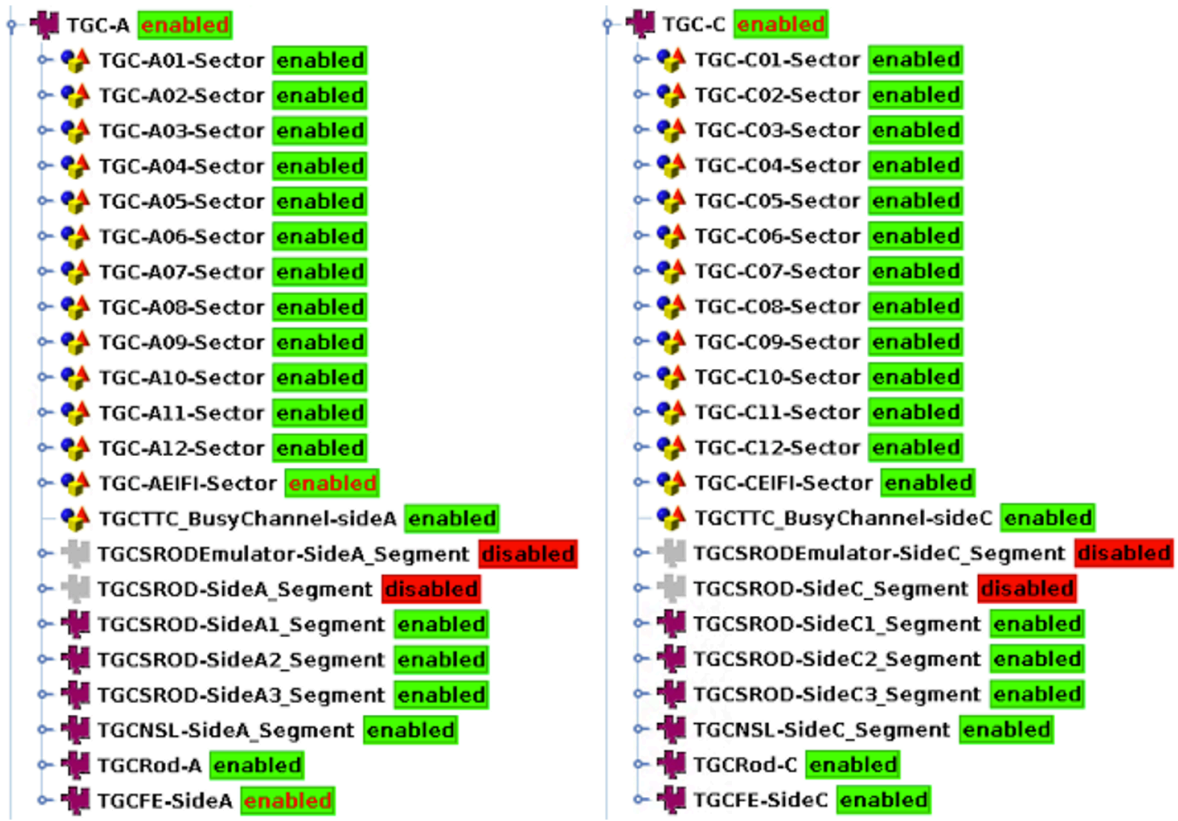


図 C.3　TGC-A および TGC-C Segment の構造。

る。TGC-Monitoring Segment 内には TGC-MonDaq-Link を含む ResourceSetOR が用意されているので、TGC Segment を disable すると自動的に TGC-Monitoring Segment 内の Resource も disable されるような実装になっている。TGCTTC は名前通り TGC の TTC システムを制御するための Segment で、TGC-A と TGC-C はそれぞれ TGC の A-side および C-side の各エレクトロニクスを制御するための Segment である。これらの詳しい構造については後述する。TGC_DQ や TGCdbMonitoring はモニタリングのために用意されている Segment であるが、ここでは説明を省略する。

まず、図 C.3 に TGC-A および TGC-C Segment の構造を示す。これらは同じ構造を持つので、ここでは TGC-A に注目してそれらを説明する。TGC-A の直下に位置する Segment は各エレクトロニクスを統合制御するために用意されている。名前が TGCSROD から始まる Segment は SROD を走らせるために新たに導入したものであり、その内部構造は図 6.4 に示した。TGCNSL-SideA_Segment は New SL クレートを制御するためのものであり、その内部構造は図 5.13 に示した。これらについては本論で説明したので、本小節では改めて説明しない。

TGCRod-A と TGCFE-SideA はともに Run 2 でも使用されていた Segment で、それぞれ ROD クレートと CCI クレートを制御する。他方で TGC-A の直下に位置する ResourceSet や Resource は主に enable/disable の伝播を実現するために用意されているものである。TGC-AXX-Sector (XX=01–12) という ResourceSet は、TGC の 1/12 セクター単位で関係するエレクトロニクスを enable/disable するための ResourceSet である。TGC-AEIFI-Sector も同様に、EI/FI に関係するエレクトロニクスを一気に enable/disable するための ResourceSet である。TGCTTC_BusyChannel-sideA は TGC-A Segment の enable/disable を TGCTTC Segment 内の A-side の LTP に伝播するための Resource である。以下ではこれらの各 Segment と ResourceSet の内部構造について説明する。

図 C.4(a) に TGCRod-A Segment の構造を示す。TGCRod-A は 12 台の TGC ROD と 1 台の ROD Busy Module を制御するための Segment である。TGC ROD の制御のためには TGC-ROD-AXX-R (XX=01–12) という ResourceSetOR が用意されており、その配下にはデータの読み出しを行う TGCCircRead-AXX というアプリケーション、Run Control の制御を担う TGC-RCD-AXX という ResourceSetAND、そしてデータの出力先である ROS のリンクを表す ROL-TGC-EC-00-67000X (X=0–c) という Resource が含まれる。ROD Busy Module の制御のためには TGC-RODBusy-Side-A という ResourceSet が用意されており、その配下には入力を表す Resource

(a) TGCRod-A の構造　　(b) TGCFE-SideA の構造

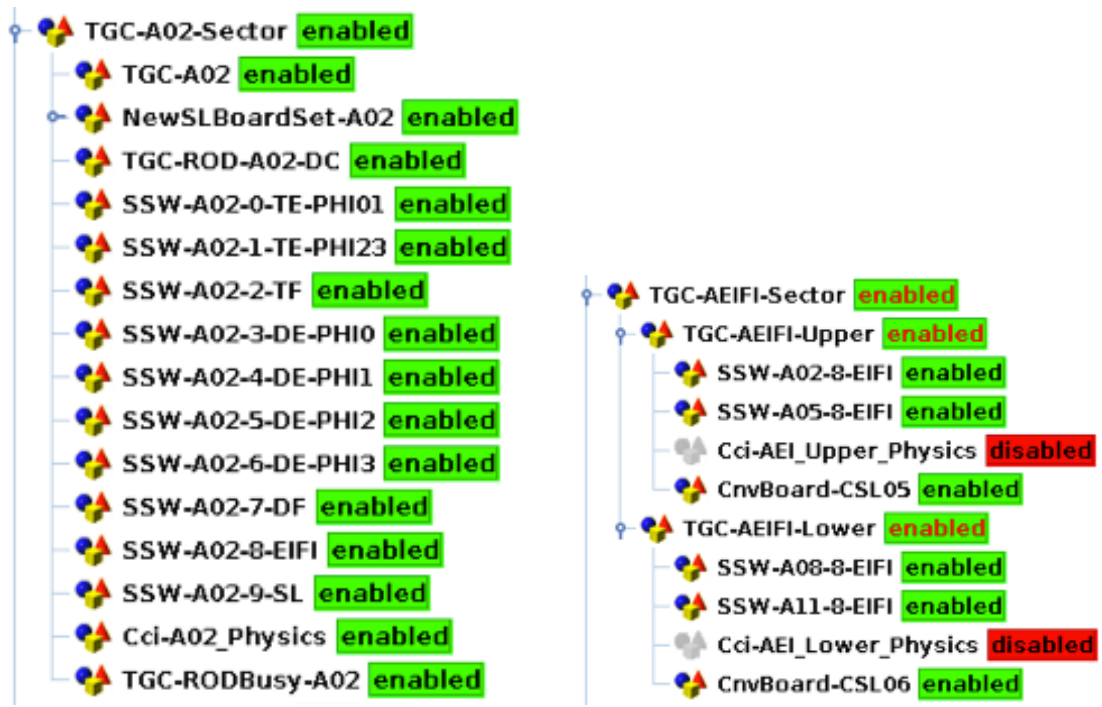
図 C.4　TGCRod-A と TGCFE-SideA Segment の構造。図 C.4(a) が TGCRod-A、図 C.4(b) が TGCFE-SideA の階層構造である。

がある。ここに TGC-TTCFOB-Busy-AX (X=1–3) という Resource を追加し、SROD PC 上の S-LINK Card が出力する busy 信号を伝播できるようにしたことについては 6.3 節で詳しく述べた。

図 C.4(b) に TGCFE-SideA Segment の構造を示す。TGCFE-SideA の配下には CCI クレート内の 4 台の SBC のそれぞれで走る RCD として、TGCFE-RCD_ECA0X (X=1–4) が用意されている。それぞれの配下には、図 5.2 の下部に示したような、TGCFE_CciModule class の Resource が含まれる。Run 2 では USA15 に設置されていた旧 SL を制御するための Resource も存在していたが、Run 3 に向けて削除した。

TGC-A Segment の直下にある TGC-AXX-Sector は前述したように、TGC の 1/12 セクター単位でエレクトロニクスを enable/disable するために用意してある ResourceSet である。その 1 つである TGC-A02-Sector の内部構造を図 C.5(a) に示す。TGC-A02、TGC-ROD-A02-DC、および SSW-A02-... はすべて ROD の制御に必要な Resource である。Cci-A02_Physics はすでに説明した通り CCI モジュールを制御するための Resource で、TGC-RODBusy-A02 は A02 セクターの ROD からの busy 信号線を表す Resource である。特に、該当する New SL に A02 セクターの enable/disable を伝播させるために、NewSLBoardSet-A02 を新たに置いた。他の 1/12 セクターの ResourceSet においても同様に NewSLBoardSet を配置することで、TGC のすべての 1/12 セクターの enable/disable が各 New SL および Run 2 まで使用していたエレクトロニクスの両方に伝播されるように実装した。

図 C.5(b) に TGC-AEIFI-Sector の構造を示す。TGC-AEIFI-Sector は A-side の TGC EI/FI に関係するエレ



(a) TGC-A02-Sector の構造　　(b) TGC-AEIFI-Sector の構造

図 C.5　TGC Sector の ResourceSet の構造。図 C.5(a) が TGC の 1/12 セクターに関係するエレクトロニクスをまとめている TGC-A02-Sector、図 C.5(b) が TGC の EI/FI に関係するエレクトロニクスをまとめる TGC-AEIFI-Sector の階層構造である。

クトロニクスをまとめる ResourceSet である。この配下には TGC EI/FI の上側 (A02 と A05 セクター) を担当する TGC-AEIFI-Upper と、下側 (A08 と A11 セクター) を担当する TGC-AEIFI-Lower という ResourceSet が含まれる。Upper および Lower にはそれぞれ該当する SSW と CCI の Resource が Run 2 のときにすでに含まれていた。4.2.3 節でも説明したように Run 3 からは G-Link Converter を使って TGC EI/FI の PS Board から来る信号をまとめるので、該当する CnvBoard をそれぞれの ResourceSet に新たに導入した。C-side でもこのような実装を行うことで、TGC EI/FI の enable/disable が各 G-Link Converter にも伝播されるようにした。

最後に図 C.2 の TGCTTC Segment の構造について説明する。図 C.6 にその階層構造を示す。TGCTTC の直下には A-side の TTC システムを制御する TGCTTC-SideA_DAQSlice と、C-side の TTC システムを制御する TGCTTC-SideC という Segment がある。この 2 つの構造は同じなので、A-side のみを説明する。TGCTTC-SideA_DAQSlice の配下に 3 つの ResourceSet と 1 つの Resource がある。TGCTTC-A_ResetSet という ResourceSet は配下に TGCFE_PsReset-A という、コンフィギュレーション時に TGC のフロントエンド回路に PS Reset 信号を送る ResourceApplication を持つ。TGCTTC_BusyChannel-sideA は TGC-A Segment の enable/disable を感知するために用意されている Resource である。TGCTTC-A_DAQSlice は TTC システムの RCD を含む ResourceSetOR で、LTP、TTCvi、Delay Module などのモジュールを制御する役割を担う。そして Run 2 からの主な変更点は、TGC-TTCFOB-A という ResourceSetAND を新たに導入したことである。A-side の TTC システムが正常に動作していない場合は A-side の TTCFOB および New SL クレートも正常に動作しないと想定するべきである。TGC-TTCFOB-A の配下には TTCFOB-A-CSL0X (X=4–6) を配置し、A-side の TTC Segment の enable/disable が A-side の 3 つの New SL クレートに伝播されるような実装にした。C-side にも同様の実装を行ない、Run 3 の新規エレクトロニクスが TGC の TTC システムとも同期して制御されるようにした。

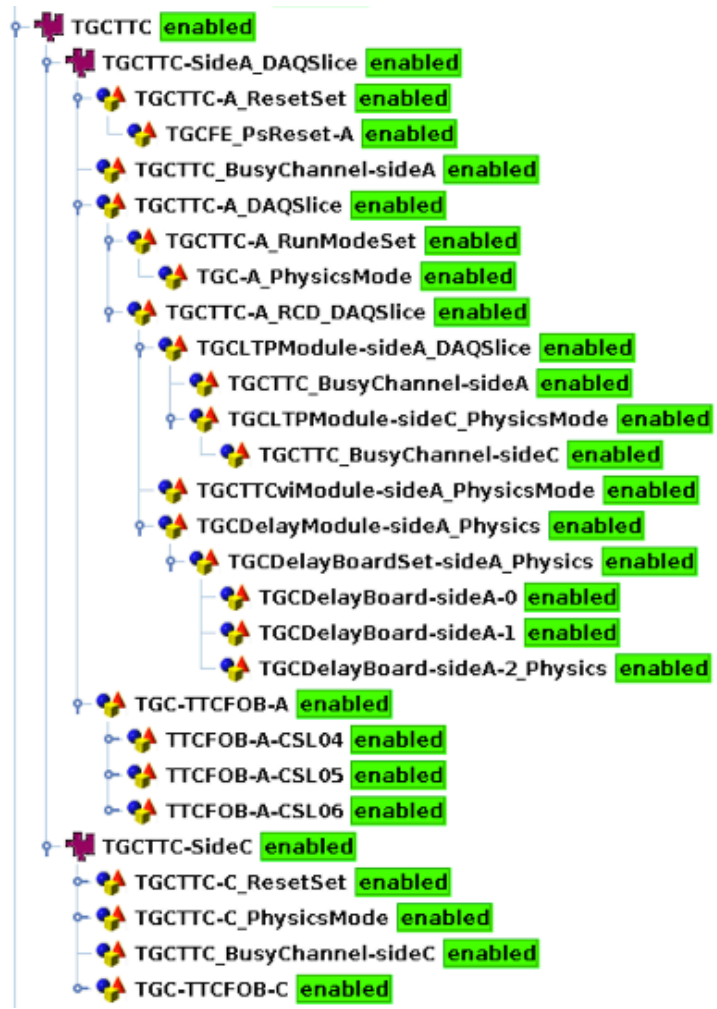


図 C.6　DAQSlice Partition の TGCTTC Segment の構造。物理ランで用いる ATLAS Partition では TTC 信号の発行元が異なるので、TGCTTC Segment 内の構造もやや異なる。

付録 D

State Transition やモニタリング機能についての補足

ここでは 5.4 節と 5.5 節で省略した事項についてまとめて補う。

D.1 TTC Restart の実装方法

本論では TTC Restart の具体的な実装方法については省略したので、その説明をここに記す。TGC Segment に対して TTC Restart を実行すると、TGC Segment の RC State は SHUTDOWN RC Command によって RUNNING から ABSENT (図 5.4 の RootController における NONE と同義) に遷移し、内部の各 Segment や Resource に INITIALIZE から START までの RC Command が自動的に送られる。対応するソフトウェアアプリケーションでは CONFIGURE、CONNECT、START で呼ばれる上位の関数が再び呼ばれることになる。すなわち TTC Restart の際はフラグを立てて、不要な操作を飛ばすという方法で実装を行なった。図 D.1 に New SL を操作する L1MuESLModule パッケージにおいて TTC Restart のフラグを立てる操作のコードを示す。TTC Restart を実行している最中の RootController の RC State は RUNNING になっているので、通常のコンフィギュレーションのときとの区別が可能である。CONFIGURE の始めで呼ばれる setup(...) という関数の中で RootController の RC State の情報を IS から取ってくることで、TTC Restart のときはフラグを立てるような実装にした。TTC Restart の際に飛ばしたい操作はこのフラグを参照する。L1MuETTCFanoutModule、L1MuEGLinkCnvModule、L1MuEBurstStopperModule にも同様の実装を行い、New SL クレートの各エレクトロニクスで TTC Restart を判別できるようにした。

D.2 I^2C の具体的な通信手順

本論では I^2C 通信の手順は簡略化して説明したが、ここでは具体的な手順について説明する。図 D.2 に沿って SDA と SCL の 2 本の接続線を用いた通信手順について説明する。一般的な通信手順は以下の通りである：

1. まず通信を始める前に SDA と SCL がともに high である必要がある。両方 high の状態からマスターが SDA のみを low にすることで通信が始まる。
2. SCL が low のときのみ SDA の状態を操作して送信するビットを指定することができる。
3. SCL を high にすることでスレーブが SDA のビットを取り込む。
4. 2. と 3. を 8 回繰り返し、8 ビットの情報をマスターからスレーブに送る。
5. スレーブが 8 ビットの情報を取り込むことができたら、SDA 線に low の acknowledge (ACK) 信号を返す。マスター側で ACK を読むことで 8 ビット分の情報を送れたかの確認ができる。SCL を上げ下げすれば ACK

```
555 void L1MuESLModule::setup(DFCountedPointer<ROS::Config> configuration)
556 {
557   NewSL_LOG_MESSAGE("enter");
558
559   // introduced TTC restart (2020.3)
560   NewSL_LOG_MESSAGE("Obtaining Run Control State...");
561   ISInfoDictionary isInfoDict(ipcPartition_);
562   std::string current_state = "UNKNOWN";
563   try {
564     RCStateInfo stateinfo;
565     isInfoDict.getValue("RunCtrl.RootController", stateinfo);
566     current_state = stateinfo.state;
567   } catch(daq::is::Exception& ex) {
568     NewSL_ERROR_MESSAGE("Failed to read IS info 'RunCtrl.RootController' : " << ex.what());
569   }
570   NewSL_LOG_MESSAGE("Current Run Control state is: " << current_state);
571   if(current_state == "RUNNING") {
572     NewSL_INFO_MESSAGE("L1MuESLModule configuration is currently in TTC restart mode.");
573     m_isTtcRestarted = true;
574     NewSL_LOG_MESSAGE("m_isTtcRestarted = " << m_isTtcRestarted);
575   }
```

in L1MuESLModule/src/L1MuESLModule.cxx

図 D.1　L1MuESLModule での TTC Restart の実装。TTC Restart の際は RootController の RC State が RUNNING になっている。この情報を IS から取得することで、通常のコンフィギュレーションのときとの区別ができる。

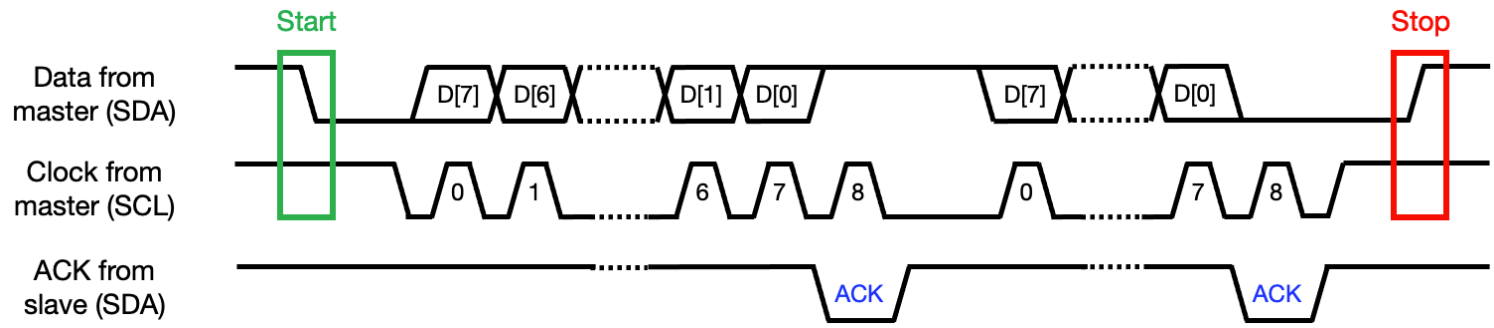


図 D.2　I^2C の通信手順。

は消える。

6. 2. から 5. を繰り返し、8 ビット単位でデータを読み書きする。
7. 通信を終了する際は SCL が low の状態で SDA を high にする。

I^2C 通信ではこのようにして 8 ビット単位でデータの読み書きを行う。どのような手順で読み書きを行うかはそのモジュールが採用しているロジックに依存する。5.5 節で述べたように、TTCrx と SFP+ 光トランシーバの間でもスレーブの使い方が異なるので、それぞれに適合した API を別々に開発した。

D.3　TTCrx の I2C_ID の特定方法

TTCrx と通信を行うためには、その固有の 6 ビットの I2C_ID を知っていなければならないことは 5.5.2 節で述べた通りである。しかし TTCrx にリセットをかけると、チップ上の配線によって決まっているデフォルトの I2C_ID が表 5.4 に示したレジスタに保有される。このデフォルトの I2C_ID がわからない場合は I^2C 通信が行え

ず、レジスタの値を読み出すことはできない。そのため TTCrx の I2C_ID を見つける機能を搭載したテストコマンド (testTtcrxI2ConTTCFOB) を開発し、USA15 に設置されている全 6 台の TTCFOB の TTCrx のデフォルトの I2C_ID を割り出した。表 D.1 のそれらの値を示す。これらは表 C.2 に示した TTCrxI2cId という attribute として OKS で管理し、オンラインソフトウェアを用いて参照できるようにした。

表 D.1　USA15 の TTCFOB 上の TTCrx の 6 ビット I2C_ID

TTCFOB のクレート ID	TTCrx の I2C_ID
CSL01	44
CSL02	37
CSL03	9
CSL04	7
CSL05	15
CSL06	0

付録 E

高速読み出しシステムの開発の補足

ここでは 6 章で省略した事項を補う。

E.1 トリガーデータ読み出しフォーマットの補足：New SL の入出力データ

New SL が SROD に送信する読み出しデータのフォーマットは 6.4.1 節ですでに説明したが、その詳細は省略した。New SL が送信する読み出しデータは各検出器エレクトロニクスから受信したトラック情報と、MUCTPI へと出力するトリガー判定情報から構成される。ここではこれらのデータのフォーマットについて補足する。

E.1.1 New SL の入力データ：各検出器エレクトロニクスから受信するトラック情報

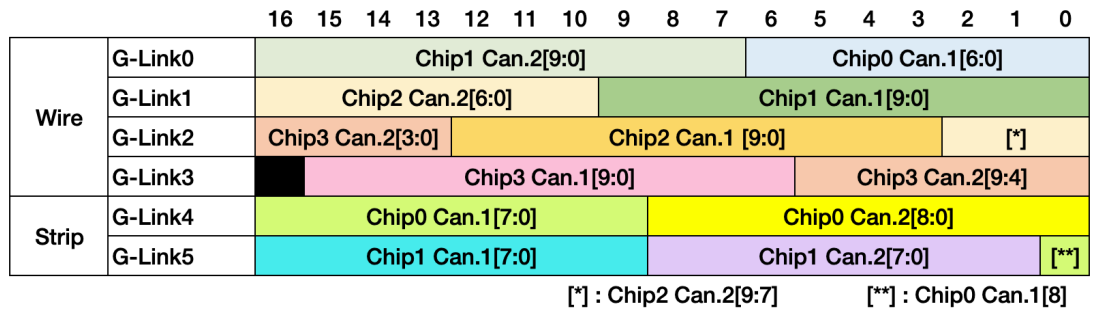
New SL は TGC BW、TGC EI/FI、NSW、RPC BIS78、および Tile Calorimeter から送られてくるトラック情報を基にトリガー判定を行う。まずはこれらの入力データのフォーマットについて説明する。

New SL が受信する TGC BW のトラック情報

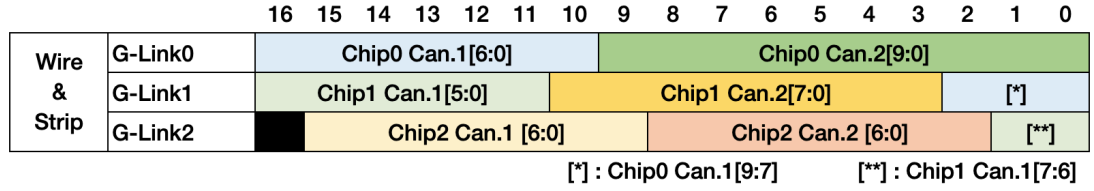
4.2.3 節でも説明したように、エンドキャップセクター用の New SL (エンドキャップ New SL) およびフォワードセクター用の New SL (フォワード New SL) はそれぞれ 2 つの TGC トリガーセクター (4.1.1 節参照) を担当する。トリガーセクター 1 つあたりのトラックの最大数はエンドキャップとフォワードで異なるので、別々に説明する。

エンドキャップ New SL が受信する TGC BW のトラック情報のデータフォーマットを図 E.1(a) に示す。トリガーセクター 1 つあたり 6 つの G-Link レーンを使用して、ワイヤーについては最大で 7 本のトラック情報を、ストリップについては最大で 4 本のトラック情報を受信する。これにより、1 つのトリガーセクターについて送られてくる最大のデータ量は 1 BC あたり 101 ビットである。すなわち 1 台のエンドキャップ New SL は 1 BC あたり 202 ビットで、最大 $(7+4)\times 2 = 22$ 本の TGC BW のトラック情報を受信することになる。

フォワード New SL が受信する TGC BW のトラック情報はエンドキャップ New SL の場合と比べて少ない。そのデータフォーマットを図 E.1(b) に示す。トリガーセクター 1 つあたり 3 つの G-Link レーンを使用して、ワイヤーとストリップ合わせて最大 6 本のトラック情報を受信する。これにより、1 つのトリガーセクターについて送られてくるデータ量は 1 BC あたり 50 ビットであり、1 台のフォワード New SL は 1 BC あたり 100 ビットで最大 12 本の TGC BW のトラック情報を受信する。



(a) エンドキャップ New SL が受信するデータ



(b) フォワード New SL が受信するデータ

図 E.1　New SL が受信する TGC BW のトリガーセクター 1 つあたりのトラックデータのフォーマット。図 E.1(a) はエンドキャップ New SL が受信するデータのフォーマットである。エンドキャップ New SL は 6 つの G-Link レーンで、ワイヤーについては最大で 7 本のトラック情報を、ストリップについては最大で 4 本のトラック情報を受信する。図 E.1(b) はフォワード New SL が受信するデータのフォーマットである。フォワード New SL は 3 つの G-Link レーンで、ワイヤーとストリップ合わせて最大 6 本のトラック情報を受信する。7–10 ビットの各トラック情報はヒットの位置情報、p_{T} の大小を表すビット、および $\mathrm{d}R_{13}$ または $\mathrm{d}\phi_{13}$ から構成される。

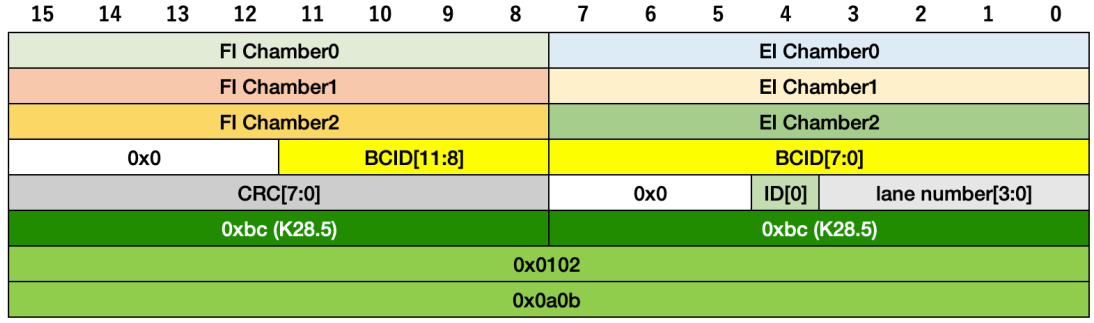


図 E.2　エンドキャップ New SL が受信する TGC EI/FI のトラックデータのフォーマット。1 台のエンドキャップ New SL は 3 つの EI/FI チェンバーの情報を受信する。各チェンバーの 8 ビットの情報はワイヤーやストリップの OR 情報から構成される。なお CRC[*2]はデータ転送に伴うエラーを検出するために用意されているものである。

エンドキャップ New SL が受信する TGC EI/FI のトラック情報

TGC EI/FI のトラック情報は G-Link Converter からエンドキャップ New SL にのみ送られる。エンドキャップ New SL はその情報を 1 つの GTX レーンを用いて受信する。そのデータフォーマットを図 E.2 に示す。1 台のエンドキャップ New SL は担当するトリガーセクターの EI/FI チェンバーに加え、その両隣のチェンバーを合わせた計 3 つのチェンバーの情報を受信する。結果的に、1 台のエンドキャップ New SL は 1 BC あたり 128 ビットの TGC EI/FI の情報を受信する。

New SL が受信する NSW のトラック情報

NSW のトラック情報はエンドキャップ New SL とフォワード New SL の両方に送信される予定で、図 E.3 に示すようにそのデータフォーマットも共通している。1 台の New SL は 6 つの GTX レーンのそれぞれで、1 BC あたり最大で 4 本のトラック情報を受信する。1 つの GTX レーンあたり 128 ビットの情報が送られてくるので、各

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
comma								comma							
track0[15:8]								track0[7:0]							
track1[7:0]								track0[23:16]							
track1[23:16]								track1[15:8]							
track2[15:8]								track2[7:0]							
track3[7:0]								track2[23:16]							
track3[23:16]								track3[15:8]							
BCID[11:4]								BCID[3:0]				ID[3:0]			

図 E.3　エンドキャップおよびフォワード New SL が受信する NSW のトラックデータのフォーマット。1 つの GTX レーン、1 BC あたり 24 ビットのトラック情報が最大で 4 つ送られてくる。各トラック情報は (η, ϕ) の位置情報やトラックの角度情報を表す $\mathrm{d}\theta$ などから構成される。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
comma								comma							
track0[15:8]								track0[7:0]							
track1[7:0]								track0[23:16]							
track1[23:16]								track1[15:8]							
track2[15:8]								track2[7:0]							
track3[7:0]								track2[23:16]							
track3[23:16]								track3[15:8]							
CRC[7:0]								BCID[7:0]							

図 E.4　エンドキャップ New SL が受信する RPC BIS78 のトラックデータのフォーマット。NSW の場合と同様、1 つの GTX レーン、1 BC あたり 24 ビットのトラック情報が最大で 4 つ送られてくる。各トラック情報は (η, ϕ) の位置情報、トラックの角度情報を表す $(\mathrm{d}\eta, \mathrm{d}\phi)$、RPC BIS78 内の 2/3 コインシデンスを表すフラグなどから構成される。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCID[3:0]				Mod3[2:0]			Mod2[2:0]			Mod1[2:0]			Mod0[2:0]		

図 E.5　エンドキャップ New SL が受信する Tile Calorimeter のトラックデータのフォーマット。1 つの G-Link レーンで 1 BC あたり 16 ビットのトラック情報が送られてくる。

New SL は 1 BC あたり $128 \times 6 = 768$ ビットの NSW のトラック情報を受信する。

エンドキャップ New SL が受信する RPC BIS78 のトラック情報

RPC BIS78 のトラック情報はエンドキャップ New SL のみが受信する。図 E.4 に示すように、RPC BIS78 のトラックデータのフォーマットは NSW のものとほぼ同じである。1 台のエンドキャップ New SL は 1 つの GTX レーンを用いて 1 BC あたり 128 ビットの RPC BIS78 のトラック情報を受信する。

エンドキャップ New SL が受信する Tile Calorimeter のトラック情報

Tile Calorimeter のトラック情報もエンドキャップ New SL のみが受信する。図 E.5 にそのデータフォーマットを示す。TMDB から各 New SL に 3 ビットのモジュール情報が最大で 4 つ送られてくる。これに BCID の下位 4 ビットが加わり、1 台のエンドキャップ New SL は 1 つの G-Link レーンを用いて 1 BC あたり 16 ビットの Tile Calorimeter のトラック情報を受信する。

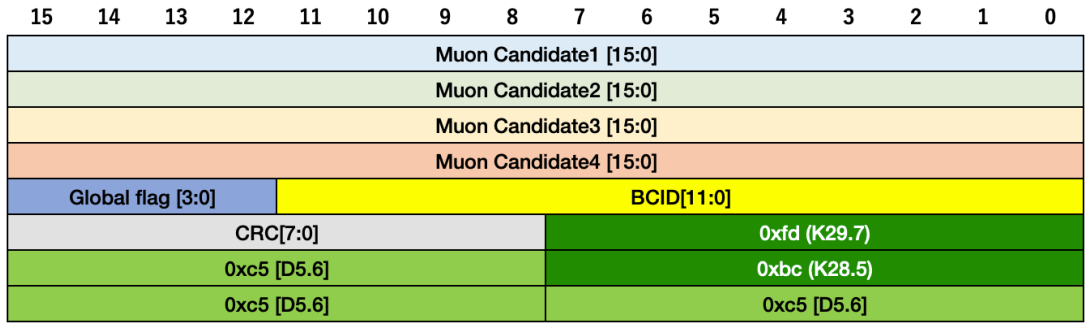


図 E.6　New SL が MUCTPI に送信するトリガー判定データのフォーマット。各 New SL は 1 つのトリガーセクターあたり、1 BC あたり最大で 4 つのトラック候補の情報を MUCTPI に送信する。16 ビットのトラック候補はコインシデンスのフラグ、p_{T}、および RoI の情報を含む。

E.1.2　New SL の出力データ：MUCTPI に送信するトリガー判定情報

各 New SL は担当する 2 つのトリガーセクターのそれぞれについて、図 E.6 に示すフォーマットのトリガー判定情報を MUCTPI に送信する。1 BC あたり最大 4 つのミューオントラック候補の情報に他の情報が加わり、最終的に 128 ビットのトリガー判定情報が送られる。すなわち 1 台の New SL が MUCTPI に送るデータ量の合計は 1 BC あたり 256 ビットとなる。

E.2　L1A レートが低い場合に生じる読み出しパスの遅延の原因調査の詳細

6.5.2 節で説明したように、L1A 出力頻度が低い場合における読み出しパスの遅延が生じる原因を解明するべく、L1A レートを 400 Hz に設定して SROD ソフトウェアが行う処理の時間測定を行なった。まずは EventBuilder アプリケーションの処理時間を調べたところ、リングバッファから EB FIFO にデータを移動させる処理に $O(100\ \mathrm{ms})$ もの時間を要していることがわかった。しかしこの処理にはリングバッファにデータが来るのを待つ時間も含まれているため、上流にある Collector アプリケーションの処理時間も調べた。すると、Collector ではソケットバッファにデータが来るのを待つ処理で $O(100\ \mathrm{ms})$ の時間を要していることが判明した。図 E.7 に TTCCollector にて検知した読み出しパスの遅延時間を記す。TTCCollector では、ソケットバッファにデータが来るのを待つ処理で約 180 ms もの時間を要していることが判明した。このことに加え、データの受信はイベントごとではなく、73 イベント分に相当する 1460 バイトのデータをまとめて受信していることもわかった。検算すると、

$$\frac{1}{400\ \mathrm{Hz}} \times 73 = 183\ \mathrm{ms} \tag{E.1}$$

より、確かに 1460 バイト分のデータをため込んだことによって遅延が生じていると説明できる。さらに SLCollector も同様に、1460 バイトのデータをまとめて受信しているということを確認した。1 イベントごとの読み出しデータ量は TTCFOB で 20 バイト、New SL で 12 バイトに設定していたので、TTCCollector と SLCollector で異なるイベント数のデータをまとめて受け取っていることがわかった。このことから、読み出しパスにおいて SROD ソフトウェアの Collector より上流で、1460 バイトのデータをバッファする箇所があると推測できる。この推測のもとで遅延の原因を調査したところ、6.5.2 節で説明した SiTCP の Nagle アルゴリズムに行き着いた。

```
Took more than 10 ms to process an event! Outputting process time of each step.
********************************************** TTC Collector Performance ********
Total processing time 1.810e+05 us.
Receive data from socket : 1.810e+05 us.
ConvertToHostByteOrder   : 3.000e+00 us.
CheckTTCDataFormat       : 4.000e+00 us.
moveDataToRingBuffer     : 1.300e+01 us.

before checkSocket()     : 0.000e+00 us.
checkSocket()            : 1.810e+05 us.
Collector::read()        : 8.000e+00 us.
after Collector::read()  : 0.000e+00 us.
Collector::getReadByte() : 1460 bytes read
m_counter_daqRun = 1
```

ソケットバッファにデータが
来るまで 181.5 ms かかっている

1460 bytes (73 イベント分)
のデータを一気に受信している

図 E.7　Collector アプリケーションで検知した読み出しパスの遅延時間。L1A レートを 400 Hz に設定し、TTCCollector アプリケーションが行う処理の所要時間をログに出力するようにした。その結果、ソケットバッファにデータが来るまで約 180 ms もの時間がかかっていることがわかった。また、1460 バイトのデータをまとめて受信していることも判明した。何度か試験を行ない、これらの値が再現されることも確認した。

E.3　L1A レートが低い場合に生じる読み出しパスの遅延時間の短縮方法の検証

6.5.3 節で提案した解決策の 1. と 2. の検証結果についてここで補足する。Nagle アルゴリズムをオフにした場合、および MSS を 20 バイト、40 バイト、60 バイト、120 バイトにした場合でどのようなふるまいをするかを調べた。この試験では 1 台の New SL と 1 台の TTCFOB の SiTCP の設定を変更し、それぞれから 1 イベントあたり 12 バイトと 20 バイトの読み出しデータを SROD に送信するようなセットアップにした。表 E.1 にその動作試験の結果を示す。デフォルトの設定では実際の物理ランにおける L1A レートである 100 kHz で正常に動作するので、これを基準としてまずは 100 kHz での動作確認を行なった。Nagle アルゴリズムをオフにした場合および MSS を 20-60 バイトと小さくした場合は、SROD ソフトウェアにて壊れたデータを検知した。具体的には Collector アプリケーションがデータを取得する時点でイベントデータが部分的に抜けていたりしたため、後段の EventBuilder にてデータが正しく読めず、エラー状態に陥った。TCP/IP ヘッダーの大きさは 40 バイトであるため、これらの条件では 1 イベントのデータサイズに対するオーバーヘッドが大きく、100 kHz という高レートでは安定動作しないという結論に至った。MSS を 120 バイトにした場合はデフォルトの設定と同じく、正常にデータを送信することができていた。しかし L1A レートを 400 Hz にして MSS を 120 バイトにした場合、MSS がデフォルト値の 1460 バイトの場合と比べてその頻度は大幅に低下したものの、not deleted events がわずかに生じた。すなわち MSS を 60–120 バイトの間の値にすることで、100 kHz で安定動作を保証しつつ、L1A レートが低い場合において not deleted events を生じさせないようにできる可能性がある。しかし許された MSS の範囲が想定以上に小さく、MSS の最適化によって安定動作を保証することは困難であると判断した。

表 E.1　SiTCP の設定を変更した場合の動作試験の結果。実際の物理ランの L1A レートは 100 kHz なので、まずは L1A レートを 100 kHz にして動作を確認した。100 kHz で安定動作した条件でのみ 400 Hz で試験を行なった。最後の MSS=1460 バイトはデフォルトの設定だが、参考として載せておく。

Nagle	MSS [bytes]	100 kHz での動作	400 Hz での動作
Off	-	SROD ソフトウェアにて壊れたデータを検知した。	-
On	20	SROD ソフトウェアにて壊れたデータを検知した。	-
On	40	SROD ソフトウェアにて壊れたデータを検知した。	-
On	60	SROD ソフトウェアにて壊れたデータを検知した。	-
On	120	正常に走った。	約 30 分間走らせたところ、69.3 万イベントのうち 162 イベントが not deleted となった。
On	1460	正常に走った。	Not deleted events が生じ、15 分で ROS から XOFF が来た。

参考文献

[1] P.A. Zyla et al. (Particle Data Group), *The Review of Particle Physics (2020)*, Prog. Theor. Exp. Phys. 2020, 083C01 (2020). Web

[2] ICRR The Univ. of Tokyo, ダークマターとは？, 2015. Web

[3] S. D. M. White, C. S. Frenk, and M. Davis, *Clustering in a neutrino-dominated universe*, Astrophysical Journal, vol. 274, L1–L5, 1983. Web

[4] S. P. Martin, *A Supersymmetry Primer*, arXiv:hep-ph/9709356, 2016. Web

[5] N. Arkani-Hamed, S. Dimopoulos, and G. Dvali, *The Hierarchy Problem and New Dimensions at a Millimeter*, Phys. Lett. B 429 263–272, 1998. Web

[6] L. Randall and R. Sundrum, *A Large Mass Hierarchy from a Small Extra Dimension*, Phys.Rev.Lett. 83 3370—3373, 1999. Web

[7] ATLAS Collaboration, *Search for new phenomena in high-mass diphoton final states using* 37 fb^{-1} *of proton-proton collisions collected at* $\sqrt{s} = 13$ *TeV with the ATLAS detector*, arXiv:1707.04147, 2017. Web

[8] ATLAS Collaboration, *Search for TeV-scale gravity signatures in high-mass final states with leptons and jets with the ATLAS detector at* $\sqrt{s} = 13$ *TeV*, arXiv:1606.02265. Web

[9] CERN, *The High-Luminosity LHC (HL-LHC)*, 2020. Web

[10] ATLAS EXPERIMENT—PUBLIC RESULTS, *Annual plots : 2018 pp Collisions*, 2018. Web

[11] ATLAS Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, 2008 JINST 3 S08003, 2008. Web

[12] ATLAS Collaboration, *Operation and performance of the ATLAS semiconductor tracker*, arXiv:1404.7473, 2014. Web

[13] K. Potamianos on behalf of the ATLAS Collaboration, *The upgraded Pixel detector and the commissioning of the Inner Detector tracking of the ATLAS experiment for Run-2 at the Large Hadron Collider*, arXiv:1608.07850, 2016. Web

[14] J. Wotschack, *ATLAS Muon Chamber Construction Parameters for CSC, MDT, and RPC chambers*, ATL-MUON-PUB-2008-006, 2008. Web

[15] ATLAS Collaboration, *Update of Technical Design Report*, 2000. Web

[16] 藏重 久弥, *ATLAS TGC*. Web

[17] 救仁郷 拓人, LHC-ATLAS 実験 Run-2 に向けた Level-1 ミューオントリガーアルゴリズムとデータ収集システムの改良, 修士論文, 2015. Web

[18] ATLAS Collaboration, *Operation of the ATLAS trigger system in Run 2*, arXiv:2007.12539, 2020. Web

[19] 赤塚 駿一, LHC-ATLAS 実験 Run-3 に向けたミューオントリガーの改良, 修士論文, 2017. Web

[20] ATLAS TGC Collaboration, *Amplifier-Shaper-Discriminator ICs and ASD Boards*, 1999. Web

[21] 田代 拓也, ATLAS 実験における新しいミューオントリガー回路の開発と実装, 修士論文, 2013. Web

[22] ATLAS Collaboration, *ATLAS Endcap Muon Trigger Read Out Driver – TGC ROD*, 2007. Web

[23] ATLAS Collaboration, *LTPi Module Documentation*, 2013. Web

[24] Ph. Farthouat and P. Gallno, *TTC-VMEbus INTERFACE Rev. 1.6*, 2000. Web

[25] T. Kawamoto et al., *New Small Wheel Technical Design Report*, 2013. Web

[26] A. Ventura on behalf of the ATLAS Collaboration, *ATLAS Muon Trigger Performance*, 2019. Web

[27] S. Akatsuka et al., *Performance estimation of the Level-1 Endcap muon at Run 3*, 2018. Web

[28] K. Ntekas, *Performance characterization of the Micromegas detector for the New Small Wheel upgrade and Development and improvement of the Muon Spectrometer Detector Control System in the ATLAS experiment*, CERN-THESIS-2016-019, 2016. Web

[29] ATLAS Collaboration, *Technical Design Report for the Phase-II Upgrade of the ATLAS Muon Spectrometer*, ATLAS-TDR-026, 2017. Web

[30] L. Massa on behalf of the ATLAS Muon Collaboration, *The BIS78 Resistive Plate Chambers upgrade of the ATLAS Muon Spectrometer for the LHC Run-3*, ATL-MUON-SLIDE-2020-070, 2020. Web

[31] 岡崎 佑太, LHC-ATLAS 実験 Run-3 に向けたミューオントリガーの改良とハードウェアへの実装, 修士論文, 2018. Web

[32] 内田 智久, SiTCP 説明書 第 1.3 版, 2012. Web

[33] Supermicro, *SuperServer 5019P-WTR*, 2020. Web

[34] COMMSCOPE, *DATA SHEET Ruckus ICX 7450*, 2020. Web

[35] O. Boyle, R. McLaren, and E. v.d. Bij, *The S-LINK Interface Specification*, 1997. Web

[36] 特殊電子回路株式会社, Kintex-7 PCIe 光ボード 「Cosmo-K」, 2018. Web

[37] J. Christiansen, A, Marchioro, P. Moreira, and T. Toifl, *TTCrx Reference Manual Version 3.9*, 2004. Web

[38] SFF Committee, *SFF-8431 Specification for SFP+ High Speed Electrical Interface*, 2013.

[39] Avago Technologies, *AFBR-709SMZ Data Sheet*, 2013. Web

[40] 竹田 康亮, LHC-ATLAS 実験 Run-3 におけるデータ読み出しシステムの開発と不感領域削減のためのトリガーアルゴリズムの性能評価, 修士論文, 2018. Web

[41] 徳永 孝之, LHC-ATLAS 実験 RUN3 に向けた新しいミューオントリガー装置の FPGA 読み出し開発とその性能評価, 修士論文, 2015. Web

[42] T. Uchida, *Hardware-Based TCP Processor for Gigabit Ethernet*, IEEE Transactions on Nuclear Science, Vol. 55, No. 3, 2008. Web

[43] W. Vandelli, *Readout*, 2016. Web

[44] A. Ruiz and E. v. d. Bij, *Implementation of S-LINK Using Physical Layer at 2.5 Gbps*, 2002. Web

[45] G. L. Alberghi, *Performance of the ATLAS RPC first level Muon trigger during Run-II data taking*, ATL-MUON-PROC-2018-007, 2018. Web