

HL-LHC ATLAS ピクセル検出器量産時の品質試験に向けた データベースシステムの構築

東京工業大学 理学院物理学系物理学コース 陣内研究室
奥山広貴 (19M00398)

2021 年 1 月 12 日

Abstract

The ATLAS Experiment is being conducted with the Large Hadron Colider at CERN. Its purposes are measurement of the Standard Model (SM) and searches for particles beyond the SM.

In order to acquire more statistics and to achieve more advanced measurements and searches, LHC is plannig to increase the luminosity, referred to as HL-LHC. The target luminosity and integrated luminosity is approximately seven times and ten times higher respectively.

Due to the upgrade, it is required for detectors to have more radiation tolerance and high granularity. It is planed to replace the ATLAS inner detector to the new one, referred to as the Inner Tracker(ITk). ITk entirely consists of silicon detestors and covers much more wider solid angle acceptance than the current Inner Detector.

For the production of ITk, we are planning to produce $O(10,000)$ modules and conduct a series of tests for quality control(QC tests) for individual modules. Those are carried out repeatedly in the production flow. All the QC tests should be stored to a central database, which is set up at Unicorn university in Czeck Republic, to record the performance of modules itself as the reference for the operation of the ITk.

“Local database” system have been developed in a previous study in order to manage data at local production sites and to synchronize informations to check the central DB. The system have been under test-use among production sites towards full production. There were items left to be developed for the module production. Particularly the devlopment of the funcsions specialized for the module production and QC tests and the synchronizing tools between the central database and local database.

I have developed the database system in this study. I have implemented key functions for the production, for example searching results, synchronizing data between the local and the central database, and validated that we can use the whole functionalities of the system, including my developed tools, using devices at the laboratory. Additionally I confirmed that we can use the tools with the actual data at the production to measure the processing time of the services.

概要

欧州原子力研究機構 (CERN) に設置されている大型ハドロン衝突型加速器 (LHC) の 1 つの衝突点にて、LHC-ATLAS 実験が行われている。この実験は現在、素粒子物理学の基本理論となっている標準模型の精密測定や標準模型を超えた新粒子の探索を目的としている。

更なる測定、探索に向けて取得統計数の増加を狙い、LHC では 2025 年より加速器をアップグレードし、衝突確率に対応する量である瞬間ルミノシティをあげる計画を予定しており、これを HL-LHC と呼ぶ。瞬間ルミノシティは現在の LHC の約 7 倍、積分ルミノシティは約 10 倍となる予定である。

HL-LHCにおいて、検出器には高い放射線耐性や位置分解能の向上など現在のものよりも高い水準が要求される。そのため、ATLAS 実験では最内層に設置している内部飛跡検出器の総入れ替えを予定しており、新しく製造する検出器を Inner Tracker (ITk) と呼ぶ。ITk では全ての領域でシリコン検出器が搭載され、また現在の内部飛跡検出器よりも広い立体角をカバーする設計となっている。

ITk の製造に向けてピクセルモジュール約 10,000 台を生産し、各モジュールに対して品質試験を行う予定となっている。品質試験は項目が多く、モジュール組み立て工程の中で何度も行うものである。モジュール情報及び品質試験の結果は固体性能の保持や ITk 運転時における参照を目的としてチェコに設置されている中央データベースに保存する必要がある。

また各組み立て機関においてモジュール及び品質試験の情報管理に使用することを目的とした「ローカルデータベースシステム」が、先行研究で開発されており、いくつかの機関でシステムの試験運用が行われていた。このシステムはモジュールの量産及び品質試験での使用に向けていくつかの開発課題が残されていた。特に品質試験に特化した機能開発や中央データベースとローカルデータベース間の同期機能開発は、量産時には必要となる機能であるが実装されていなかった。

本研究では、これらのデータベースシステム構築を行なった。ローカルデータベースにおける品質試験管理機能の 1 つとして結果検索機能やデータベース間の同期機能を開発し、システムの拡張を行なった。また品質試験におけるデータベース操作の流れを確立し、データベース機能全体が流れの中で使用可能であることを確認した。さらに開発した機能が、量産時に想定されるデータに対して十分に使用可能であるかを、機能の処理時間測定を行い確認した。

目次

概要	i
第 1 章 序論	1
1.1 素粒子物理と標準模型	1
1.2 LHC について	1
1.3 ATLAS 実験	1
1.4 HL-LHC 実験アップグレード計画	6
第 2 章 ピクセルモジュール	10
2.1 ピクセルモジュールの構造	10
2.2 ピクセルモジュールの構成部品	10
2.3 新型モジュールの種類	14
第 3 章 検出器量産と品質試験	15
3.1 組み立て工程	15
3.2 品質試験	15
3.3 検出器量産におけるデータ管理	26
第 4 章 モジュール情報及び品質試験結果管理システム	27
4.1 中央データベース	27
4.2 ローカルデータベース	29
4.3 本研究における開発項目の詳細	35
第 5 章 品質試験のデモンストレーション	49
5.1 デモンストレーションと機能確認	49
第 6 章 ローカルデータベースにおける検索機能とその処理時間	63
6.1 実装方法	63
6.2 処理時間測定	70
6.3 改善方法の処理時間測定	73
第 7 章 中央データベースとローカルデータベースの同期	78
7.1 サーバーの設置場所による処理時間の違い	78
7.2 モジュール情報のダウンロード	81

7.3	読み出し試験結果のアップロード	89
第 8 章	まとめ	96
8.1	本論文のまとめ	96
8.2	現状と今後の課題	97
付録 A	シリコン検出器の原理	99
A.1	半導体 [8]	99
A.2	検出原理	101
付録 B	RD53A の回路図とフレキシブル基板	102
付録 C	ローカルデータベースのチュートリアルと普及状況	104
付録 D	モジュール生産状況の解析	106
付録 E	ファイル送信時におけるデータ容量と処理時間の関係について	107
参考文献		110
謝辞		113

¹ 第1章

² 序論

³ 欧州原子力研究機構 (**CERN**) に設置されている大型ハドロン衝突型加速器 (**LHC**) では、現在、素
⁴ 粒子物理学の基礎となっている標準模型の精密測定や標準模型を超える物理現象の探索が行われている。
⁵ ATLAS 実験は LHC 上にある 4 つの衝突点の 1 つで行われている実験であり、ATLAS 検出器を用いて
⁶ 崩壊粒子の測定が行われている。LHC では加速器のアップグレード (**HL-LHC**) を予定しており、これ
⁷ に向けて ATLAS 検出器のアップグレードを行う。この章では LHC-ATLAS 実験とそのアップグレード
⁸ 計画について説明する。

⁹ 1.1 素粒子物理と標準模型

¹⁰ hoge

¹¹ 1.2 LHC について

¹² LHC は CERN の地下およそ 100 m に設置されている周長 26.7 km の大型ハドロン衝突型加速器である。
¹³ バンチと呼ばれる陽子のかたまりを 7 TeV まで加速し、衝突させる。世界最大エネルギーの加速器
¹⁴ である。

¹⁵ 陽子ビームの加速は 4 つの前段加速器を用いて行う。始めに水素ガス中の水素原子から電子を分離す
¹⁶ ることで陽子を生成する。その後最初の線形加速器 (Linear Accelerator: LINAC)、陽子シンクロトロ
¹⁷ ンブースター (Proton Synchrotron Booster: PSB)、陽子シンクロトロン (Proton Synchrotron: PS)、
¹⁸ スーパー陽子シンクロトロン (Super Proton Synchrotron) で加速されたのち LHC に入射する。CERN
¹⁹ にある加速器の概要を図 1.1 に示す。LHC には 4 つの衝突点があり、それぞれ ALICE(A Large Ion
²⁰ Collider Experiment)、LHCb、CMS(Compact Muon Solenoid)、ATLAS(A Toidal LHC Apparatus)
²¹ 実験が行われている。それぞれの衝突点には崩壊粒子の飛跡やエネルギーを測定するための検出器が設置
²² されており、取得したデータを元に多様な物理解析が行われている。

²³ 1.3 ATLAS 実験

²⁴ 初めに ATLAS 実験に用いる座標系と用語について説明する。まず衝突点を原点として定義しており、
²⁵ ビーム軸を z 軸、これに対して垂直な平面を $x - y$ 平面とする。 x 軸方向は原点からみて LHC リングの
²⁶ 中心に向かう方向であり、 y 軸は上に向かう方向である。方位角 ϕ は z 軸周りの角度であり、極角 θ は z

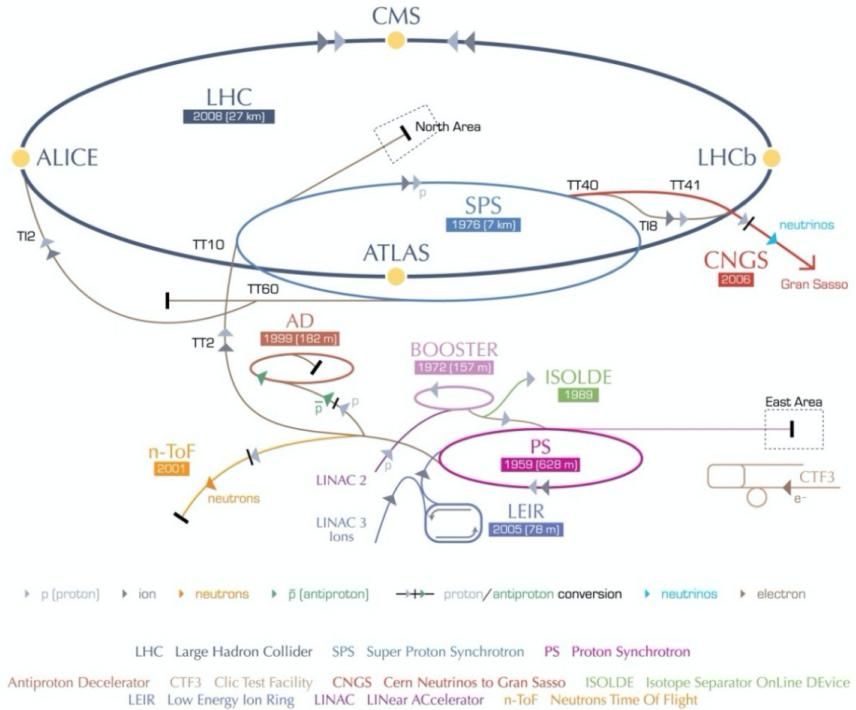


図 1.1 加速器の全体像 [1]。図は CERN に設置されている加速器の全体像を示す。陽子はいくつかの前段加速器で段階的に加速され LHC に入射する。LHC 上には 4 つの衝突点が存在し、それぞれ ALICE、LHCb、CMS、ATLAS 実験が行われている。

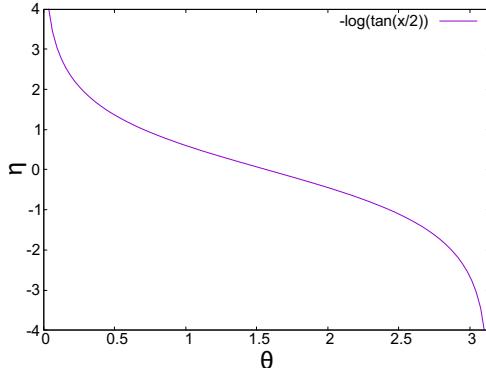


図 1.2 極角 θ と擬ラピディティ η の関係図。横軸は式 1.1 における極角 θ ($0 \leq \theta \leq \pi$)、縦軸は擬ラピディティ η を表している。図のように、 η への変換により定義域が実数全体に広がる。

²⁷ 軸とのなす角である。ATLAS 実験では、極角 θ は以下のように擬ラピディティ η で表される。また極角 ²⁸ θ と擬ラピディティ η の関係図を図 1.2 に示す。

$$\eta = -\ln \left(\tan \left(\frac{\theta}{2} \right) \right) \quad (1.1)$$

²⁹ 1.3.1 ATLAS 検出器

³⁰ ATLAS 検出器は複数の検出器から構成され、陽子の衝突によって生成された粒子の運動量、エネルギーを測定することができる。最内層に内部飛跡検出器が設置されていて、次に超電導ソレノイド磁石、カロリメータ、トロイド磁石、ミューオン検出器の順に設置されている。ビームパイプ以外をほとんど検

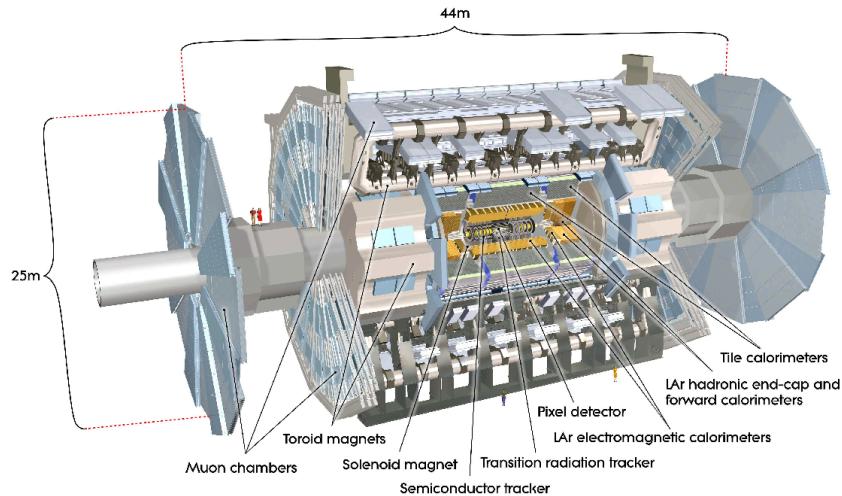


図 1.3 ATLAS 検出器の全体像 [2]。内側から内部飛跡検出器、ソレノイド磁石、カロリメータ、トロイド磁石、ミューオン検出器が設置されている。

33 出器で覆うような設計となっている。ATLAS 検出器の全体図を図 1.3 に示す。

34 1.3.2 内部飛跡検出器

35 ATLAS 検出器の最内層に位置する検出器である。内部飛跡検出器は粒子の飛跡測定をするための検出
36 器であり、飛跡の曲率から粒子の運動量を計算できる。検出器の外側には超伝導ソレノイド磁石が設置さ
37 れており、2 T の磁場が z 方向にかけられる。これにより荷電粒子はローレンツ力を受け、軌跡が曲がる。
38 内部飛跡検出器は 3 つの検出器で構成され、内側からピクセル検出器、ストリップ検出器、遷移放射検
39 出器の順に設置されている。ピクセル、ストリップ検出器は階層構造になっており、粒子は複数の検出器
40 を通過する。それぞれの検出器で取得した通過位置をつなぎ合わせることで粒子の軌跡を計算するこ
41 ができる。

42 内部飛跡検出器の全体図を図 1.4 に示す。

43 ピクセル検出器

44 内部飛跡検出器の最内層に位置する検出器である。ピクセル検出器はバレル部が 4 層、エンドキャップ
45 部が 6 層で構成される。バレル部の最内層は IBL(Insertable B-Layer) と呼ばれ、順に B-Layer、Layer-1、
46 Layer-2 となっている。

47 ピクセル検出器の全体図を図 1.5 に示す。

48 ピクセル検出器の各層は、モジュールと呼ばれる最小単位の検出器をいくつも搭載している。ピクセル
49 モジュールを図 1.6 に示す。このピクセルモジュールの詳細については 2 章で述べる。

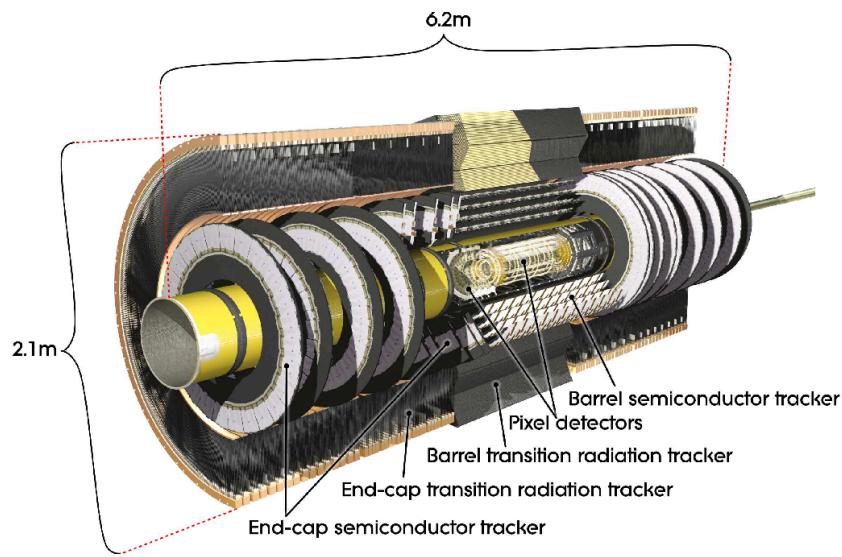


図 1.4 内部飛跡検出器の全体像 [2]。内側からピクセル検出器、ストリップ検出器、遷移放射検出器が設置されている。また円筒状のバレル部とディスク上のエンドキャップ部を構造として持つ。

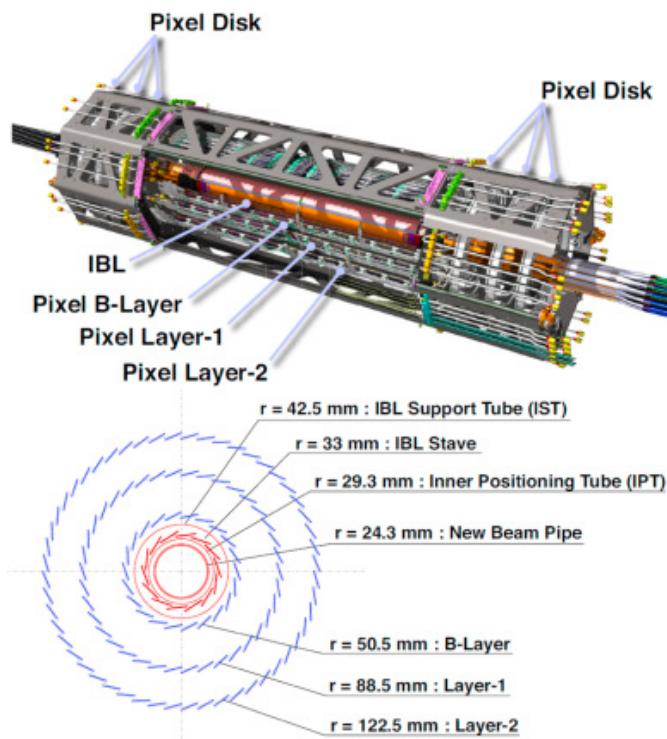


図 1.5 ピクセル検出器の全体像 [3]。上図はピクセル検出器の全体を模式的に表したものであり、下図はビーム軸方向から見たピクセル検出器バレル部の断面図である。バレル部の 4 層は内側から IBL、B-Layer、Layer-1、Layer-2 と呼ぶ。

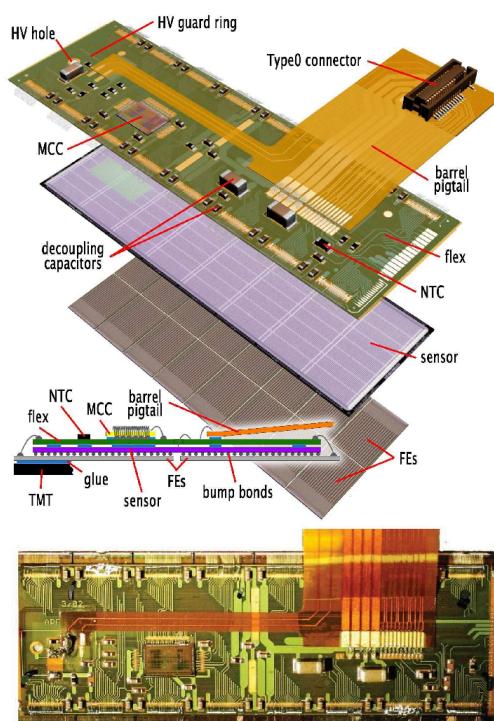


図 1.6 ピクセルモジュールの全体像 [2]。ピクセル検出器は、最小単位としてモジュールと呼ばれる構造を持ち、図にモジュールの全体像を示している。モジュールは、荷電粒子が通過し、信号を生成するセンサー部、AD 変換を行う FE チップ部、データ転送を行うフレキシブル基板から構成される。

表 1.1 現行 LHC と HL-LHC の比較 [4]。瞬間ルミノシティは 7 倍、積分ルミノシティは 10 倍になることが見積もられており、取得統計数の増加が期待できる。

	LHC	HL-LHC
重心系エネルギー	14	14
瞬間ルミノシティ [$\text{cm}^{-2}\text{s}^{-1}$]	1×10^{34}	7×10^{34}
積分ルミノシティ [fb^{-1}]	300	3,000
1 陽子衝突あたりのイベント数	27	140



図 1.7 HL-LHC 運転計画 [5]。2025 年の初めより HL-LHC の導入、2027 年の途中より本運転が始まる。

50 1.4 HL-LHC 実験アップグレード計画

51 LHC では加速器のアップグレードを予定しており、これを HL-LHC アップグレード計画と呼ぶ。詳細
52 を以下に示す。

53 1.4.1 概要

54 HL-LHC ではルミノシティ呼ばれる陽子バンチの密度を上げることで、衝突頻度を大きくし、取得統
55 計数を増やす目的がある。LHC と HL-LHC の比較を表 1.1 に示す。

56 LHC の運転計画を表 1.7 に示す。2025 年の初めより HL-LHC の導入が始まり、2027 年の途中から
57 HL-LHC 運転開始の予定となっている。

58 1.4.2 内部飛跡検出器のアップグレード

59 ルミノシティの増加に伴い、検出器には以下のようない性能が要求される。

- 60 ● 放射線耐性の向上
- 61 ● 高速読み出し

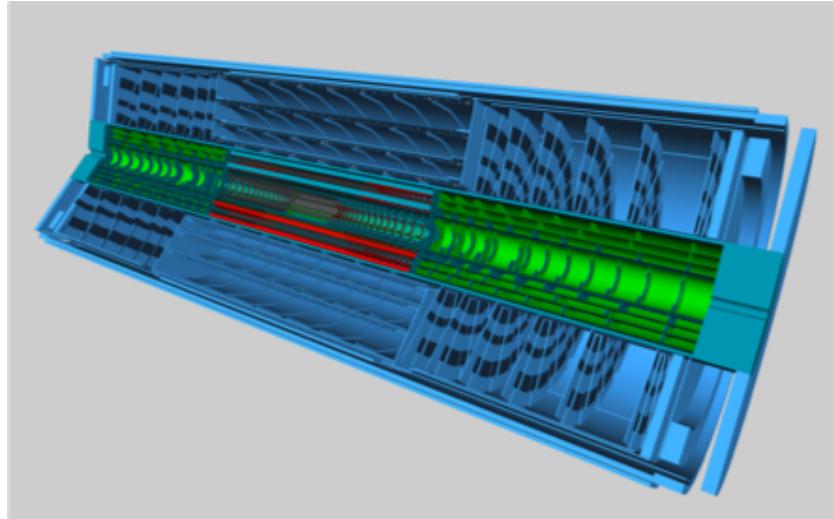


図 1.8 ITk の全体像 [6]。図は ITk 全体の模式図を示している。ITk はピクセル検出器（緑の領域）とストリップ検出器（青の領域）から構成される。

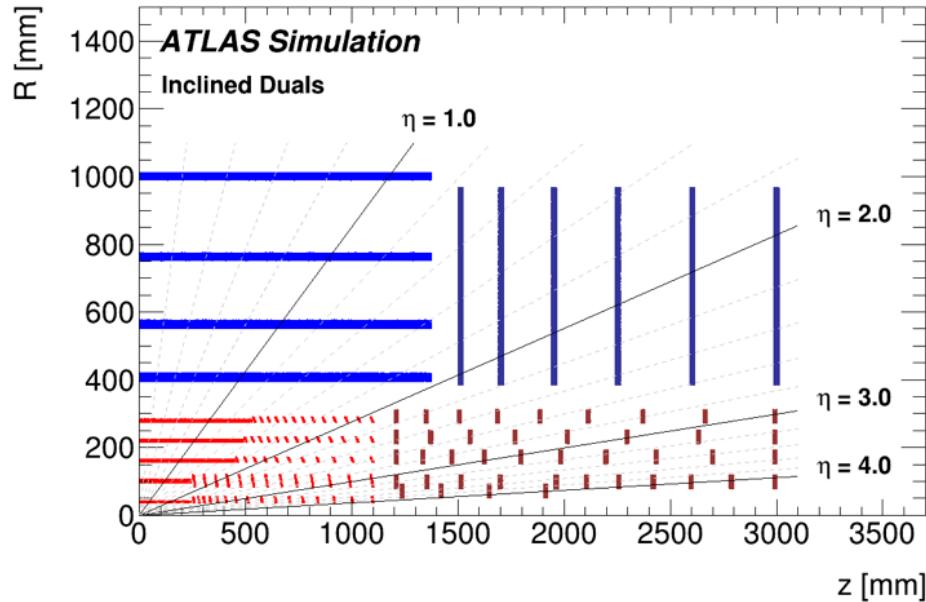


図 1.9 ITk の断面図 [6]。ITk はピクセル検出器とストリップ検出器から構成され、バレル部はそれぞれ 5、4 層の構造を持つ。ピクセル検出器はバレル部、傾斜バレル部、エンドキャップ部に分かれ、このエンドキャップ部の設計により $|\eta| < 4.0$ までの範囲をカバーし、粒子の飛跡測定をすることができる。

62 ● 検出器の細密化

63 HL-LHC 向けて ATLAS 内部飛跡検出器はアップグレードを予定しており、検出器の総入れ替えを行なう。アップグレード後の検出器を **Inner Tracker (ITk)** と呼ぶ。模式図を図 1.8 に示す。

65 ITk の構成と現行ピクセル検出器との比較

66 図 1.9 に ITk のビーム軸方向の断面図を示す。ITk はピクセル検出器とストリップ検出器で構成される。ピクセル検出器はバレル、傾斜バレル、エンドキャップ部で構成され、バレル部は 5 層となっている。

表 1.2 ピクセル検出器設置領域の比較。表は現行と ITk におけるピクセル検出器設置領域の比較を示す。 η の範囲に関して、括弧内は度数法における対応する値を示す。ITk では遷移放射検出器を使用しないため、ピクセル検出器の半径方向のカバー領域が増え、層の数は 4 から 5 となる。また前方方向に生じる粒子測定を行うために、 η の領域も拡大している。

	現行	ITk
$r[\text{mm}]$	33~129	39~279
層の数 (バレル部)	4	5
$ \eta $	$< 2.5 (< 19^\circ)$	$< 4 (< 4.2^\circ)$

表 1.3 搭載するピクセルモジュール数の比較。表は現行と ITk のピクセル検出器において、各構造における搭載ピクセルモジュールの数を示している。ITk ではバレル部で現行の約 2 倍になっているのに加え、傾斜バレル部、エンドキャップ部に多くのモジュールを搭載する設計となっていることが分かる。

層	バレル部		傾斜バレル部		エンドキャップ部	
	現行	ITk	現行	ITk	現行	ITk
1	280	192	—	512	—	64
2	286	240	—	520	—	242
3	494	660	—	660	—	320
4	676	960	—	1040	288	352
5	—	1300	—	1300	—	468
合計	1736	3352	0	4032	288	1446

68 ピクセル検出器の配置に関して、現行と ITk の比較を表 1.2 に示す。またモジュール数の比較を表 1.3
 69 に示す。ITk では現行に比べ、 η が大きい領域まで検出器を設置する設計となっており、その分使用する
 70 モジュール数も増える。

71 1.4.3 物理測定に及ぼす影響

72 カバーする η の範囲が大きい特徴が生む利点として、前方方向に大きな運動量を持つ粒子を含む物理現
 73 象の測定精度向上があげられる。

74 この例として、ボゾン粒子結合によるヒッグス粒子生成過程 (Vector boson fusion higgs production,
 75 VBF) をあげる。VBF の 1 つのチャンネルに関するファインマン図及びイベントディスプレイを図 1.10
 76 に示す。この衝突により生じる 2 つのクオークはジェットと呼ばれる粒子群となり、前方方向に大きい運
 77 動量を持つ。ITk ではこれらのジェットを正確に捉えることができ、VBF イベントの測定精度を向上す
 78 ることができる。この測定における系統誤差は表 1.4 のように見積もられる [6]。

79 広い η 範囲を検出できることによる利点として、他にも b タグ精度の向上、主崩壊イベント由来の
 80 ジェット識別精度の向上、 τ レプトン識別精度の向上が見込まれ、多くの物理現象において測定精度の向
 81 上が見込まれている。

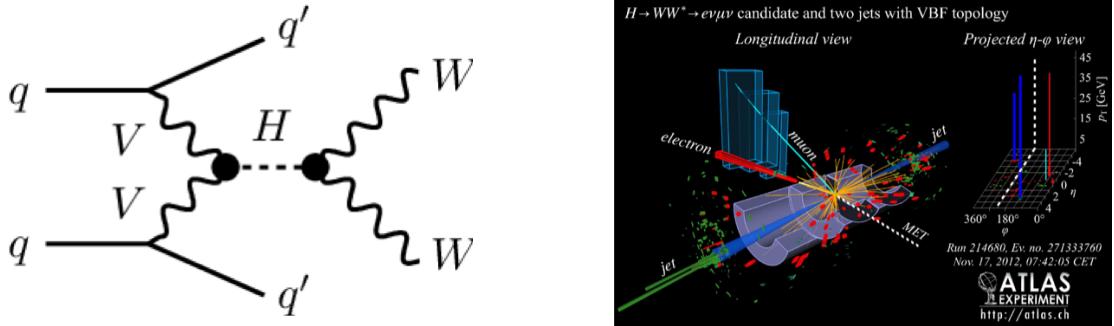


図 1.10 VBF イベントの図。図は VBF イベントの一つである $H \rightarrow WW$ 崩壊チャンネルを示している。左図はファインマン図、右図はそのイベントディスプレイ [7] を表す。このイベントにおいて、初めに相互作用したクオーク対は、それぞれ崩壊を繰り返しジェットと呼ばれる粒子群となる。このときこれらのジェットは前方方向に大きな運動量を持つ。ITk では $|\eta| < 4.0$ の範囲をカバーしており、このようなジェットを捉えることができるため、測定精度が向上する。

表 1.4 VBF $H \rightarrow WW$ の測定における系統誤差の見積もり。表は統計数 3000 fb^{-1} において、 $|\eta| < 2.7$ 、 $|\eta| < 4.0$ の領域を使用した場合の VBF $H \rightarrow WW$ イベント測定における系統誤差の見積もりを示している。 $|\eta| < 4.0$ の領域を使用することで系統誤差が向上する見積もりであることが分かる。ここでは統計誤差は考慮にいれていない。

検出器の使用範囲	$ \eta < 2.7$	$ \eta < 4.0$
系統誤差	22%	12%

82 第2章

83 ピクセルモジュール

84 この章ではピクセルモジュールの構成と各構成部品について説明する。

85 2.1 ピクセルモジュールの構造

86 ピクセルモジュールはベアモジュールとフレキシブル基板より構成される。ベアモジュールは荷電粒子
87 の通過を検知し、信号を発生するシリコンセンサーと、AD 変換を行う FE チップで構成される。ベアモ
88 ジュールが持つ FE チップの数はモジュールの種類によって異なる。モジュールの構成を図 2.1 に示す。

89 2.2 ピクセルモジュールの構成部品

90 2.2.1 シリコンセンサー

91 ピクセルモジュールに搭載するセンサーはシリコン半導体を用いている。センサー内部構造として pn
92 接合を持ち、逆バイアス電圧をかけ空乏層を広げた状態で使用する [8]。この空乏層領域に荷電粒子が通
93 過すると、Bethe-Bloch の式 [9] に従い粒子はエネルギーを損失する。このエネルギー損失量に従い、電子
94 ・ホール対が生成、これを収集することで荷電粒子の通過情報をアナログ信号として取得することができる。
95

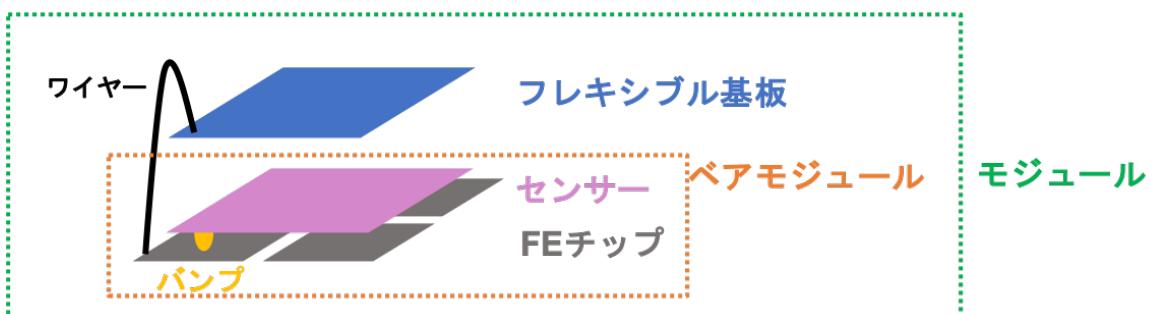


図 2.1 ピクセルモジュールの構成。図は Quad モジュールの構成を模式的に表したものである。モ
ジュールは「ベアモジュール」と呼ばれる心臓部と「フレキシブル基板」を貼り付けることで作られ
る。ベアモジュールはセンサーと FE チップから成る。Quad モジュールの場合、1 枚のセンサーに対
し FE チップは 4 枚である。センサーと FE チップは「バンプ」と呼ばれる構造で電気的に接続して
おり、1 ピクセルに対して 1 つのバンプ接合がなされている。FE チップとフレキシブル基板の接続に
は、ワイヤーが多数 ($O(100)$) 配線される。



図 2.2 新型ピクセルモジュールに搭載するシリコンセンサーの断面図。図は新型ピクセルモジュールに搭載するシリコンセンサー断面の模式図を示している。p 型半導体に n^+ 型電極を埋め込んだ $n^+ - in - p$ 型と呼ばれる構造を持つ。 p^+ 側にフレキシブル基板、 n^+ 側に FE チップが付く。逆バイアス電圧を印加すると n^+ 電極側から空乏層が広がる。

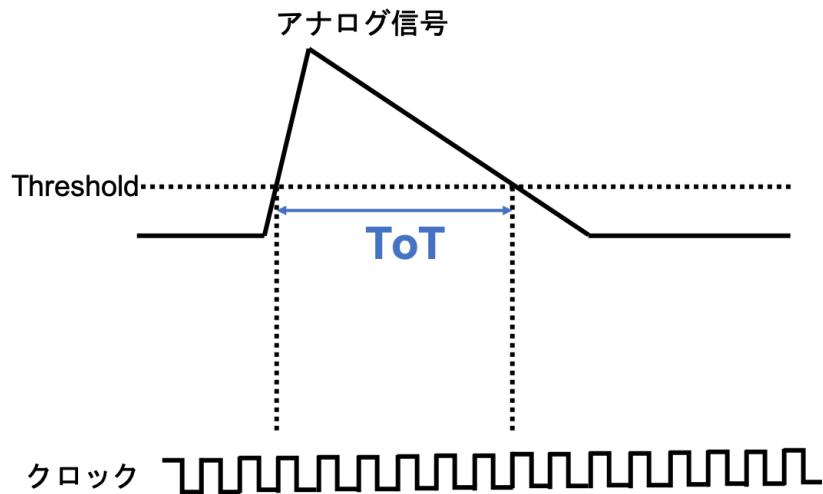


図 2.3 ToT の概念図。FE チップで生成されるデジタル信号は ToT と呼ばれる。図はその概念を模式的に表したものであり、FE チップで行われている AD 変換である。ToT はアナログ信号が Threshold 値を超えた時間幅に相当する。実際には ToT 間のクロック数 [b.c.] として取得される。ここで 1b.c.(25 ns) は LHC における陽子の衝突間隔に相当する。

96 新型ピクセルモジュールに搭載するセンサーは、 $n^+ - in - p$ 型である。模式図を図 2.2 に示す。 n^+ 電
97 極で電子を収集し、信号を取得する。現行のセンサーに比べ、型変換を起こす可能性がなくなるため安定
98 した運転が見込まれる [6]。

99 2.2.2 読み出し FE チップ

100 読み出し FE チップはシリコン半導体を用いて作られた集積回路である。読み出し FE チップの主な役
101 割は、シリコンセンサーで発生し受け取ったアナログ信号を整形、增幅したのち AD 変換し、後段に転送
102 することである。AD 変換について、アナログ信号が Threshold を超えた時間幅を測定し、デジタル信号
103 に変換する。この信号の値を **Time over Threshold (ToT)** と呼ぶ。

104 ToT の概念図を図 2.3 に示す。

105 RD53A

106 RD53A[8] は、新型ピクセルモジュールの研究、開発のために作られたプロトタイプの読み出し FE
107 チップである。RD53A の性能を表 2.1 に示す。チップサイズ、ピクセル数は、ITk に搭載予定のモジュー

表 2.1 RD53A のスペック。

チップサイズ [mm ²]	20.0 × 11.6
ピクセルサイズ [μm ²]	50.0 × 50.0
ピクセル数 [行 × 列]	400 × 192
トリガーレート [kHz]	1000
データレート [Mbps]	1280 × 4
ToT の範囲	0～15(4 ビット)
放射線耐性	500Mrad(トータルドーズ効果 [10])

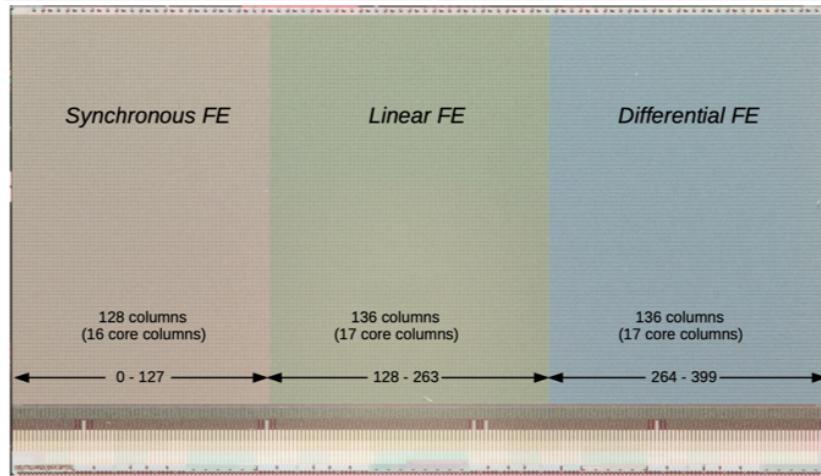


図 2.4 RD53A[8]。ピクセル数は 400 列 × 192 行となっている。図のように RD53A は 3 つの領域が設けられており、それぞれ Synchronous FE、Linear FE、Differential FE と呼ぶ。各領域のピクセルが持つアナログ回路、AD 変換回路が異なる。

¹⁰⁸ ルが持つ FE チップの半分となっている。

¹⁰⁹ FE チップ上の各ピクセルはアナログ回路部とデジタル回路部を持つ。RD53A では図 2.4 に示すよう¹¹⁰に、3 つの領域があり、それぞれの領域でピクセルのアナログ回路、AD 変換回路が異なる。左から順に¹¹¹ Synchronous FE、Linear FE、Differential FE と呼ぶ。研究、開発用に 3 つの領域が設けられている¹¹²が、性能比較の結果 ITk に搭載するモジュールには Differential FE を用いることが決定している。¹¹³なお、デジタル回路部は全てのピクセルにおいて共通である。それぞれの回路図は付録 B に示す。

¹¹⁴ 2.2.3 フレキシブル基板

¹¹⁵ フレキシブル基板は、基板上に電子部品が搭載されたものである。FE チップからのデジタル信号を後¹¹⁶段の回路へ転送する他、FE チップ、センサーへの電圧印加制御の役割も担う。

¹¹⁷ 2.2.4 信号伝達

¹¹⁸ モジュールの信号伝達の様子を模式的に表したものを図 2.5 に示す。センサーで生成した信号は FE¹¹⁹ チップ、フレキシブル基板の順に送られ、PC でデータ取得できる。

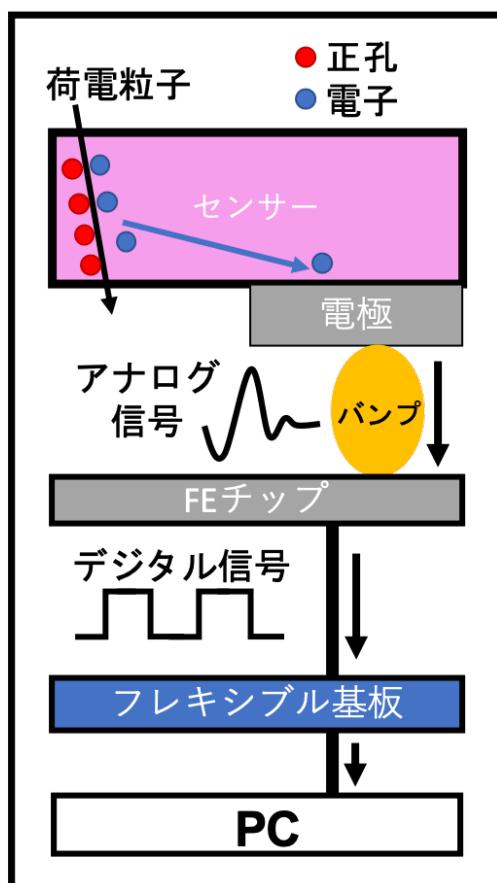


図 2.5 ピクセルモジュールにおける信号伝達の様子。図はピクセルモジュールにおける信号伝達の様子を模式的に表したものである。初めに、荷電粒子がセンサー部を通過し、エネルギー損失に応じた電子・ホール対を生成する。電子は電極により収集され、バンプを通してアナログ信号として FE チップに送られる。FE チップでは信号を整形、増幅後、デジタル信号に変換する。その後ワイヤーを通してフレキシブル基板に転送、そして PC に転送されデータを取得する。

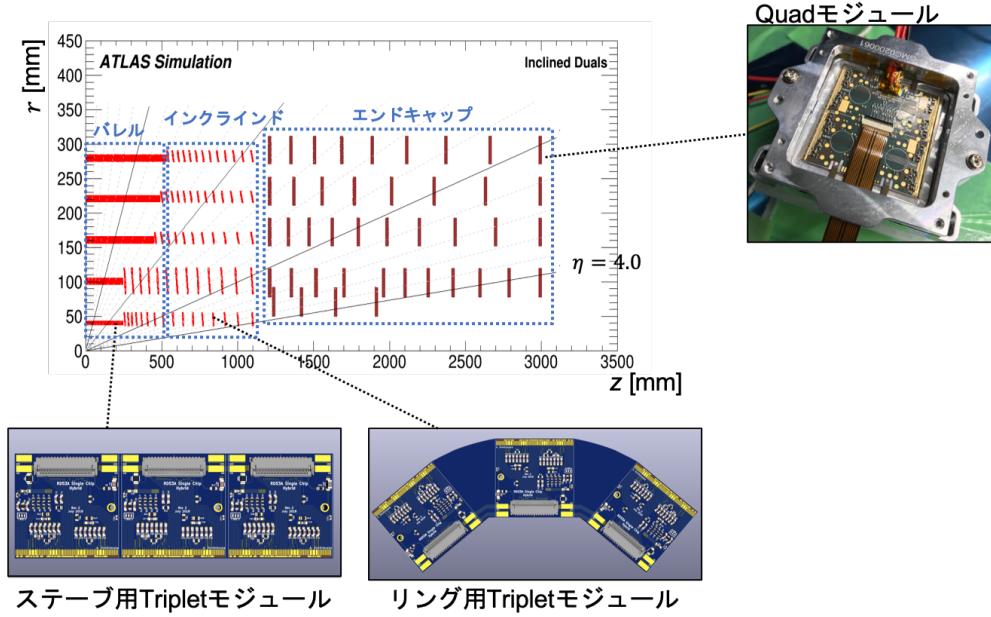


図 2.6 搭載するモジュールのプロトタイプと ITk における配置。2 種類の Triplet モジュールと Quad モジュールを搭載する予定となっている。Triplet モジュールは最内層のステーク構造、リング構造の領域に使われる。それ以外の領域には Quad モジュールが使われる。

2.3 新型モジュールの種類

現在、ITk に搭載するモジュールとして以下の 3 つのモジュールを予定している。

- ステーク用 Triplet モジュール
- リング用 Triplet モジュール
- Quad モジュール

Triplet、Quad モジュールはそれぞれ FE チップを 3、4 枚搭載するモジュールである。それぞれのモジュールの図及び検出器上における位置の一例を図 2.6 に示す。

127 第3章

128 検出器量産と品質試験

129 この章ではモジュールの組み立て工程と品質試験について説明する。

130 3.1 組み立て工程

131 モジュール組み立て機関は、初めにベアモジュールとフレキシブル基板を受け取る。組み立て工程とし
132 て以下が設定されている。

133 1. フレキシブル基板・ベアモジュール貼り付け

- 134 ● 受け取ったベアモジュールとフレキシブル基板を接着剤を用いて貼り付ける。

135 2. ワイヤー配線

- 136 ● ワイヤー配線を行い、FEチップとフレキシブル基板を電気的に接続する。

137 3. ワイヤー保護

- 138 ● ワイヤーが損傷があり断線が起きると、そのワイヤーに接続されているピクセルの読み出しが
139 できなくなる。これを防ぐため、モジュールに屋根型の構造を取り付け、ワイヤーを物理的に
140 保護する。

141 4. パリレンコーティング

- 142 ● モジュール読み出し部以外での電通や放電を防ぐため、パリレン高分子を用いてモジュールを
143 保護する。

144 5. 温度サイクル試験

- 145 ● ITk運転時の環境温度は -45°C から 40°C まで変化しうる[11]。この温度変化に耐えうるか
146 を確認するため、温度サイクルを行う。

147 6. 低温耐久試験

- 148 ● ITk運転時の典型的な環境温度は $-15\sim 0^{\circ}\text{C}$ 付近である。これに耐えうる性能を持つかを確
149 認するために、低温下にモジュールを長時間設置する耐久試験を行う。

150 流れと各組み立て工程のイメージを図3.1に示す。

151 3.2 品質試験

152 各組み立て工程に対して、いくつかの品質試験を行う。行う品質試験の代表的なものを以下に示す。

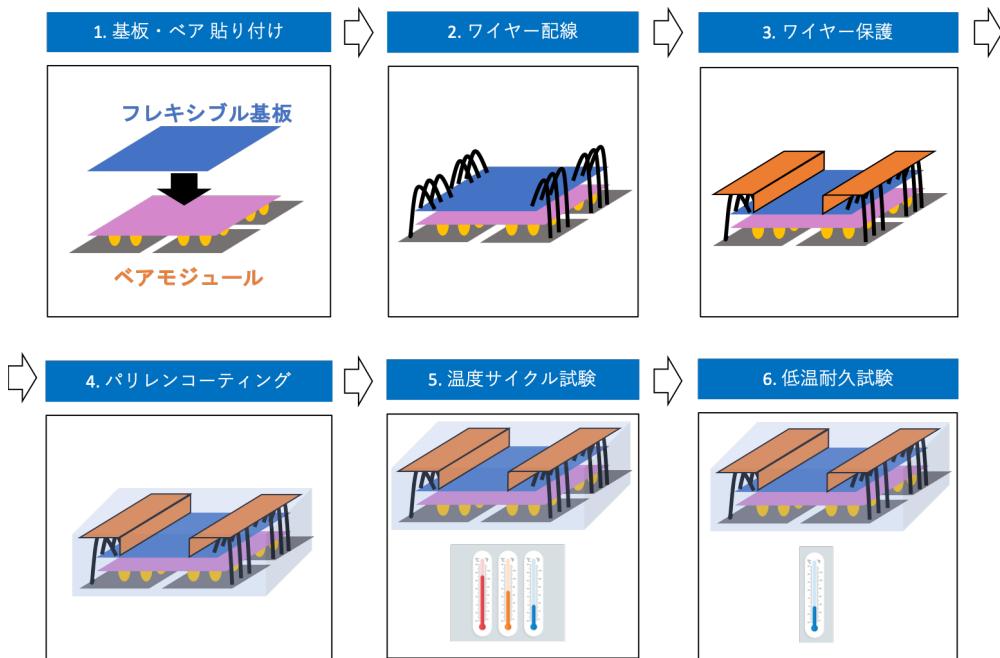


図 3.1 組み立て工程のイメージ図。モジュール組み立て機関はペアモジュール、フレキシブル基板を受け取り、それらの貼り付け、ワイヤー配線、ワイヤー保護、パリレンコーティング、温度サイクル試験、低温耐久試験の順に組み立てを行う。

153 3.2.1 外観検査

154 モジュールの外観写真を撮り、モジュールに以下のような欠陥がないかを確認する。また外観検査の様
155 子を図 3.2 に示す。

- 156 ● 抵抗等取り付け部品の損傷.
- 157 ● ワイヤーの接着位置確認.
- 158 ● 基板上の回路やワイヤーの断線.
- 159 ● 付着汚れ.

160 3.2.2 質量測定

161 モジュールの質量を測定する試験である。

162 3.2.3 平坦性測定

163 モジュール上の位置座標を何点か測定し、モジュールの平坦度、厚さ、歪み具合等を測定する。測定の
164 様子と解析の例を図 3.3 に示す。

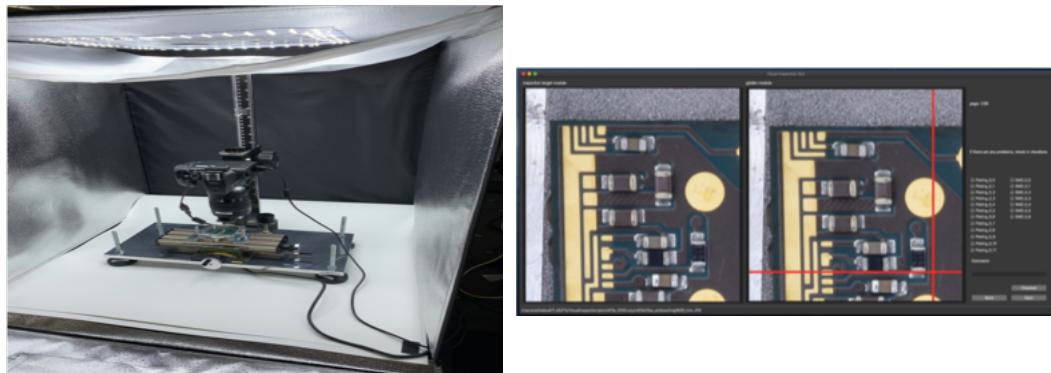


図 3.2 外観検査の様子。図は日本で実際に行われた外観検査における写真撮影の様子（左図）と撮影写真の確認画面（右図）である。左図のように、写真撮影は光量を制御するため暗箱の中で行っている。またモジュールに損傷や断線がないかを確認するために、右図のように撮影した写真と良好なモジュールの写真を見比べ、電子部品の損傷、回路の断線などの何らかの所見があった場合には記録をする。

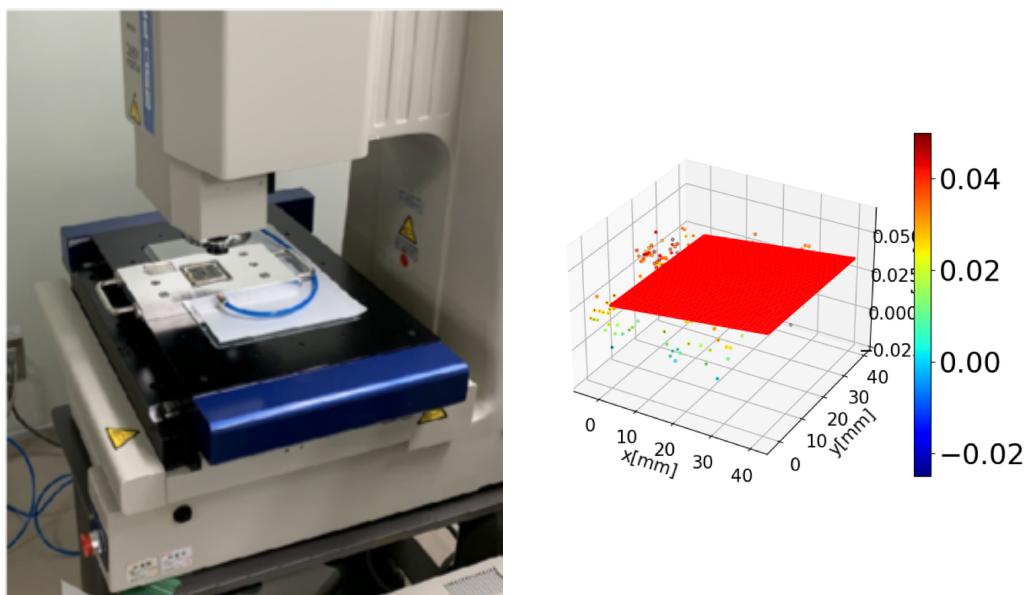


図 3.3 平坦性測定の様子。図は日本で実際に行われた平坦性測定における測定の様子（左図）と解析結果（右図）である。専用の装置を用いてモジュールの位置座標を何点か測定する。得られた測定点は右図のように図示し、平面でフィットティングを行う。フィット平面からのズレの分布や、モジュールの厚みを計算する。

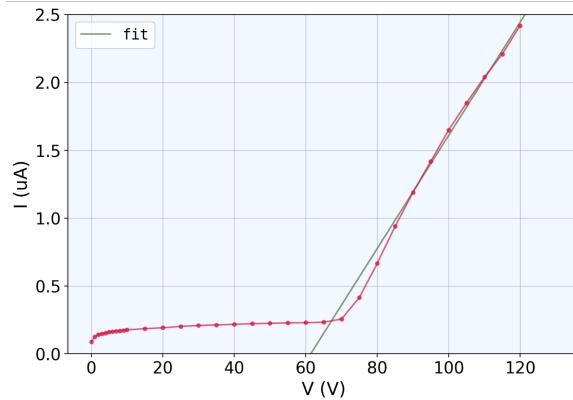


図 3.4 センサー電流-電圧特性結果の例 [12]。横軸にセンサーに対する印加電圧 [V]、縦軸に電流 [μA] を示す。逆バイアス電圧を印加しているため、電流が 0V から 60V 付近までは電流がほとんど流れていらないが、70V 付近から電流が急激に増加していることが分かる。

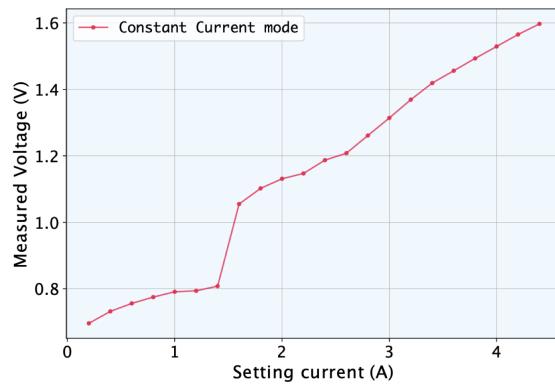


図 3.5 FE チップ電流-電圧特性試験結果の例 [12]。横軸に FE チップに対する設定電流、縦軸は測定した実際に FE チップにかかる電圧値を示している。1.5V 付近に急激に電圧が増大しており、この段階で FE チップが機能するようになる。それ以降は線形になっているのが分かる。

165 3.2.4 センサー電流-電圧特性確認

166 モジュールのシリコンセンサーに逆バイアス電圧をかけ、電流-電圧特性をみる。印加電圧を段階的に
 167 変化させて測定点をとり、電流と電圧の関係を確認する。この試験の結果の例を図 3.4 に示す。逆方向電
 168 圧では電流はほとんど流れないが、降伏電圧に達すると急激に増大する。これは pn 接合の特性 [8] であ
 169 り、正常であればこの振る舞いを確認することができる。

170 3.2.5 FE チップ電流-電圧特性確認

171 FE チップに対して電圧をかけ、電流-電圧特性をみる。センサーに対してと同様に、印加電圧を段階的
 172 に変化させて測定点をとり、電流と電圧の関係を確認する。抵抗として振る舞うため、電流、電圧間の関
 173 係は図 3.5 のように線形性を持つ。

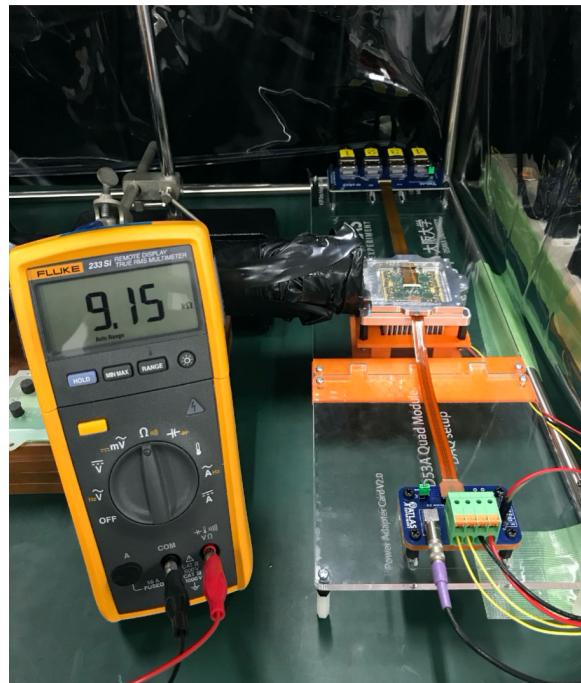


図 3.6 読み出し試験の様子 [12]。図は日本で実際に行われた読み出し試験の様子である。図の右側に見えるように、モジュールにケーブルを配線し、PCとの通信を行うことでデータ取得をする。また図の左側では抵抗値を測定している。これはモジュールに付属しているサーミスタの抵抗値を確認することで、温度情報を取得するためのものである。

174 3.2.6 読み出し試験

175 読み出し試験では、読み出し回路が正常に機能するのかを確認する。読み出し試験の様子を図 3.6 に示す。モジュールに読み出しケーブルを配線し、PC と通信を行い、データを取得する。また試験において 176 温度、FE チップやセンサーに与える電圧、電流といった検出器の操作、環境関わる情報取得を行って 177 行い、異常がないかを確認する。これらの情報は「Detector Control System, DCS」と呼ぶ。

179 汎用読み出しシステム YARR

180 YARR(Yet Another Rapid Readout) システム [13] は、ピクセルモジュール用に開発された、
181 PCI Express(PCIe) 接続を用いた読み出しシステムである。ファームウェア、ソフトウェアから構成さ
182 れる。YARR ではファームウェア上で行う処理はデータ通信やトリガー処理等の最低限に抑え、その他
183 多くの処理をソフトウェアで担うという特徴がある。

184 読み出し試験に使用するファイルと変数

185 YARR で扱う全てのファイルは JSON(JavaScript Object Notation) と呼ばれる形式で記述さ
186 れる。

187 YARR を用いた読み出し試験では以下の設定ファイルが要求される。

- 188 • 試験設定ファイル
 - 189 – 読み出し試験の初期設定や解析手法を記述する。
- 190 • ハードウェア設置ファイル

191 – 試験に用いるハードウェアの指定や設定を記述する。

192 ● 接続設定ファイル

193 – 読み出しを行う FE チップの種類やチャンネルを記述する。

194 ● FE チップ設定ファイル.

195 – 各 FE チップ毎に出力され、全ピクセルに共通な試験の設定値、各ピクセル固有の設定値を記述する。

197 また試験 1 項目ごとに以下のファイルが、1 つのディレクトリに生成される。

198 ● 試験結果ファイル.

199 – 各ピクセルの Occupancy 等、読み出し試験の結果値を記述する。

200 ● 試験設定ファイル

201 ● FE チップ設定ファイル.

202 – 試験の中で変更が加えられるため、各 FE チップにつき試験前後の 2 つのファイルが出力される。

204 ● 試験ログ

205 – 試験情報を記録する。

206 後述するローカルデータベースシステムにおいてはさらに以下の設定ファイルを用いる。

207 ● データベース設定ファイル.

208 ● 試験者設定ファイル.

209 ● 試験場所設定ファイル.

210 読み出し試験項目において以下の *Occupancy* という量が定義される。試験用電荷の入射数や発行トリガの数等、発行した信号数を n_i 回、取得信号数を n_0 としたとき、

$$\text{Occupancy} = \frac{n_0}{n_i} \times 100 [\%] \quad (3.1)$$

212 と定義する。

213 また読み出し試験では各 FE チップに対して「レジスタ」と呼ばれる設定値が存在し、FE チップ上の

214 全ピクセルに対して共通なものを「グローバルレジスタ」、各ピクセル固有のものを「ピクセルレジスタ」
215 と呼ぶ。

216 読み出し試験項目

217 以下に読み出し試験項目の一覧を示す。

218 レジスタの読み書き

219 グローバル及びピクセルレジスタの読み書きが正常にできるのかを確認する試験.

220 デジタル回路読み出し

221 各ピクセルのデジタル回路部に試験用電荷を入射し、信号の応答数を確認する試験. デジタル回路
222 部の性能確認に用いる.

223 アナログ回路読み出し

224 各ピクセルのアナログ回路部に試験用電荷を入射し、信号の応答数を確認する試験. アナログ回路
225 部の性能確認に用いる.

226 Threshold 測定

227 各ピクセルの Threshold 値を測定する試験.

228 Threshold グローバルレジスタ調整

229 全ピクセルに共通なレジスタの変更、基準となる Threshold に近づけるための調整.

230 Threshold ピクセルレジスタ調整、再調整、精密調整

231 各ピクセル固有のレジスタの変更、基準となる Threshold に近づけるための調整.

232 ToT グローバルレジスタ調整

233 全ピクセルに共通なレジスタの変更、基準となる ToT に近づけるための調整.

234 ノイズ占有率測定

235 各ピクセルのノイズの頻度を確認する試験.

236 スタックピクセル測定

237 入力電荷の有無にかかわらず、常に信号を出力するピクセルを確認する試験.

238 クロストーク測定

239 各ピクセルのクロストークの有無を確認する試験.

240 バンプ接続確認測定

241 各ピクセルのバンプ接合が正常かを確認する試験.

242 外部トリガーを用いた測定

243 外部トリガーを用いて信号の取得を行う試験. 放射線源を用いた測定に使用.

244 主な試験項目の詳細について以下で説明する。

245 デジタル回路読み出し

246 各ピクセルのデジタル回路部に試験用電荷を入射し、信号を取得する。Occupancy(式 3.1) は、入射電
247 荷数 n_i と取得信号数 n_0 で定義される。YARR のデフォルトでは $n_i = 100$ となっている。

248 デジタル回路読み出しにおける Occupancy の二次元分布の例を図 3.7 に示す。

249 この試験の結果として出力されるファイルを以下に示す。

250 OccupancyMap 各ピクセルの Occupancy を記す.

251 Enmask Occupancy = 100 のピクセルを 1、それ以外を 0 とした値を記す.

252 L1Dist 信号取得タイミングの分布を記す.

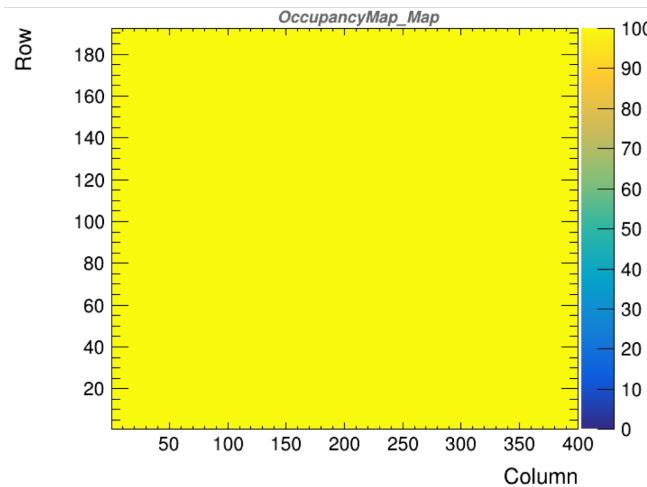


図 3.7 デジタル回路読み出しにおける *Occupancy* 分布の例。図は RD53A を用いたデジタル回路読み出しの結果であり、横軸、縦軸はそれぞれピクセルの列 (400)、行 (192) に対応する。z 軸は各ピクセルにおける *Occupancy* の値を示している。正常なピクセルは 100 付近になることが期待され、図では全てのピクセルで 100 であることが分かる。

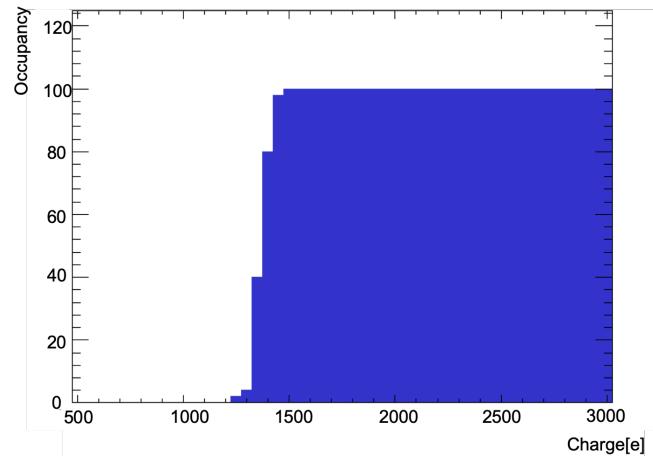


図 3.8 入射電荷量と *Occupancy* の関係。横軸はピクセルに対する入射電荷量 [e]、縦軸は *Occupancy* を示す。入射電荷量の単位 e は、電荷量 C を素電荷 $e = 1.6 \times 10^{-19}$ で割ったものである。Threshold 測定の際にはこのように入射電荷量を増加させながら *Occupancy* の値を測定する。測定結果を後述する式 (3.2) フィットし、Threshold とノイズの値を得る。

253 アナログ回路読み出し

254 各ピクセルのアナログ回路部に試験用電荷を入射し、信号を取得する。デジタル回路読み出しと同様、
 255 *Occupancy*(式 3.1) は、入射電荷数 n_i と取得信号数 n_0 で定義される。試験結果として出力されるファイルはデジタル回路読み出しと同じ形式である。

257 Threshold 測定

258 各ピクセルのアナログ回路部に試験用電荷を入射し、入射電荷数 n_i と取得信号数 n_0 より *Occupancy*
 259 を測定する。これを入射電荷量を増加させて繰り返し行う。あるピクセルにおけるこの処理結果の例を図
 260 3.8 に示す。

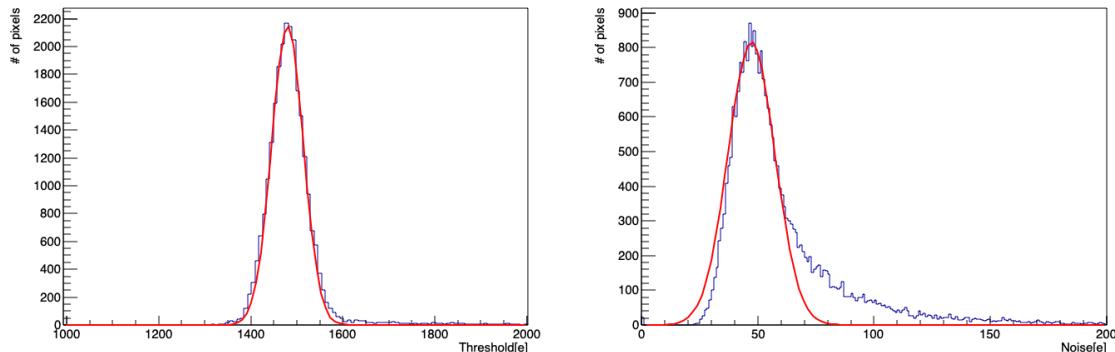


図 3.9 Threshold 値とノイズの分布。図はある FE チップ上のピクセルにおける Threshold 値 (左図) とノイズ (右図) の分布を示している。また赤線はガウス関数によるフィット関数を示しており、これにより平均値と幅を取得する。

261 この分布を以下の式でフィッティングする。フィッティングの形に由来し、これを「S カーブフィッティ
262 ング」と呼ぶ。

$$f(x) = 0.5 \times \left[2 - \text{erfc} \left(\left\{ \frac{x - Q}{\sigma \times \sqrt{2}} \right\} \right) \right] \times p \quad (3.2)$$

$$\text{erfc}(x) = 1 - \text{erf}(x) \quad (3.3)$$

263 ここで $\text{erf}(x)$ はガウスの誤差関数である [14]。

264 得られた Q 、 σ がそれぞれピクセルの Threshold 値、ノイズに相当する。ある FE チップにおける
265 Threshold 値、ノイズの分布の例を図 3.9 に示す。この分布をガウス関数でフィットすることで、全ピク
266 セルに対する Threshold、ノイズの平均値と幅が得られる。

267 この試験において、出力される結果ファイルを以下に示す。

268 ThresholdMap 各ピクセルの Threshold 値を記す。

269 ThresholdDist Threshold 値の分布を記す。

270 NoiseMap 各ピクセルのノイズを記す。

271 NoiseDist ノイズの分布を記す。

272 Chi2Map 各ピクセルの S カーブフィッティングにおける χ^2 の値を記す。

273 Chi2Dist χ^2 の分布を記す。

274 sCurve あるピクセルの入射電荷量と Occupancy の関係を記す。いくつかのピクセルについて出力する。

275 sCurveMap 全てのピクセルの SCurve を重ね合わせた値を記す。

276 ToT 測定

277 各ピクセルのアナログ回路部に試験用電荷を入射し、ToT を測定する。この操作を複数回 (デフォルト
278 では 100 回) 行い、平均値と幅を求める。出力される結果ファイルを以下に示す。

279 MeanToTMap 各ピクセルの ToT の平均値を記す。

280 MeanToTDist ToT 平均値の分布を記す。

281 SigmaToTMap 各ピクセルの ToT の分散を記す。

282 SigmaToTDist ToT 分散の分布を記す。

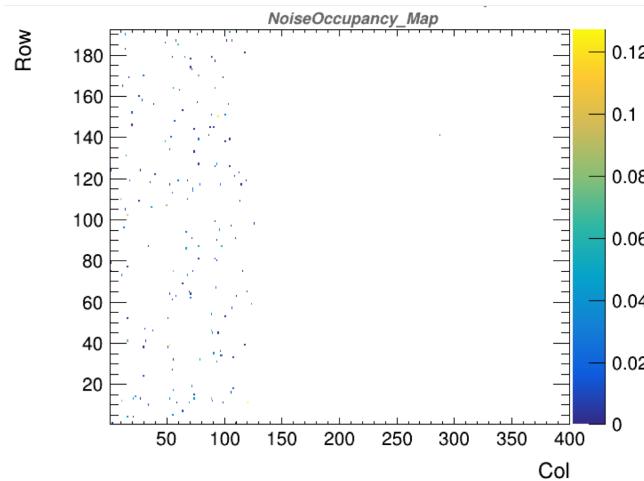


図 3.10 ノイズ占有率測定における *NoiseOccupancy* 分布の例。図は RD53A を用いたデジタル回路読み出しの結果であり、横軸、縦軸はそれぞれピクセルの列(400)、行(192)に対応する。z 軸は各ピクセルにおける *NoiseOccupancy* の値を示している。正常なピクセルでは限りなく 0 に近い値になることが期待され、図ではほとんどのピクセルが 0 となっていが、列番号が 0 から 120 付近の領域で値を大きく持つピクセルが存在する。

283 ノイズ占有率測定

284 試験用電荷を入射せずに、周波数 f [Hz]、時間 t [sec] のトリガーをかけ、取得信号数 n_0 を測定する。
285 YARR のデフォルトでは $f = 5000$ 、 $t = 300$ である。 $Occupancy$ (式 3.1) は、発行したトリガー数
286 $n_i = f \times t$ と取得信号数 n_0 で定義される。

287 また *NoiseOccupancy* を以下で定義する。*NoiseOccupancy* の分布の例を図 3.10 に示す。

$$288 \quad NoiseOccupancy = \frac{n_0}{t} \times (25 \times 10^{-9}) \quad (3.4)$$

288 25×10^{-9} [sec] = 25[nsec] は 1 b.c. である。

289 出力される結果ファイルを以下に示す。

290 OccupancyMap 各ピクセルの *Occupancy* を記す。

291 NoiseOccupancyMap 各ピクセルの *NoiseOccupancy* を記す。

292 NoiseMask $NoiseOccupancy < 10^{-6}$ のピクセルを 1、それ以外を 0 とした値を記す。

293 Threshold 調整とピクセル解析

294 今後この論文では、この試験を読み出し試験と呼ぶ。以下の流れで読み出しを行う。

- 295 • デジタル回路読み出し
- 296 • アナログ回路読み出し
- 297 • Threshold 測定
- 298 • Threshold グローバルレジスタ調整
- 299 • Threshold ピクセルレジスタ調整
- 300 • ToT グローバルレジスタ調整
- 301 • Threshold グローバルレジスタ再調整、精密調整

表 3.1 ピクセル解析の評価基準一覧 [15]。読み出し試験において各ピクセルが正常に機能しているかを判断するためにピクセル解析を行う必要があり、その判断基準が表のように定義されている。この基準一覧は不良評価であり、全ての基準に当てはまらないピクセルが健康と判断される。不良ピクセルの数や分布は、モジュールの品質を決定する1つの要素となり、ITkに搭載するモジュールや配置の決定に用いられる。

評価名	読み出し項目	評価基準
Digital Dead	Digital scan	$Occupancy < 1$
Digital Bad	Digital scan	$Occupancy < 98$ or $Occupancy > 102$
Merged Bump	Analog scan Crosstalk scan	$Occupancy < 98$ or $Occupancy > 102$ High Crosstalk
Analog Dead	Analog scan	$Occupancy < 1$
Analog Bad	Analog scan	$Occupancy < 98$ or $Occupancy > 102$
Tuning Failed	Threshold scan	Sカーブフィット失敗 (YARRでは $\chi^2 = 0$ となる)
Tuning Bad	Threshold scan ToT scan	$ Threshold - Threshold_{平均} > 5 \times Threshold_{幅}$ $ToT = 0$ or 15
High ENC	Threshold scan	$ ノイズ - ノイズ_{平均} > 3 \times ノイズ_{幅}$
Noisy	Noise scan	$NoiseOccupancy > 10^{-6}$
Disconnected Bump	Disconnected bump scan Source scan	現段階では未決定 $Occupancy$ がFEチップ全体平均の1%
High Crosstalk	Crosstalk scan	$Occupancy > 0$ with 25ke (sync FE) $Occupancy > 0$ with 40ke (lin and diff FE)

- Threshold 測定

- スタックピクセル測定

- クロストーク測定

測定後、試験結果の解析を行い、モジュール上の各ピクセルが正常かどうかを判断する。設定されてい

る評価基準を表3.1に示す。不良ピクセルには評価基準に応じた評価名が付けられる。

簡易読み出し試験

簡易読み出し試験では以下の項目を扱う。

- レジスタの読み書き
- デジタル回路読み出し
- アナログ回路読み出し
- Threshold 読み出し
- ToT 読み出し
- バンプ接続確認読み出し

バンプ接合確認試験

放射線源を用いてバンプ接合の確認を行う。以下の項目を扱う。

- バンプ接合確認測定
- ノイズ占有率測定測定

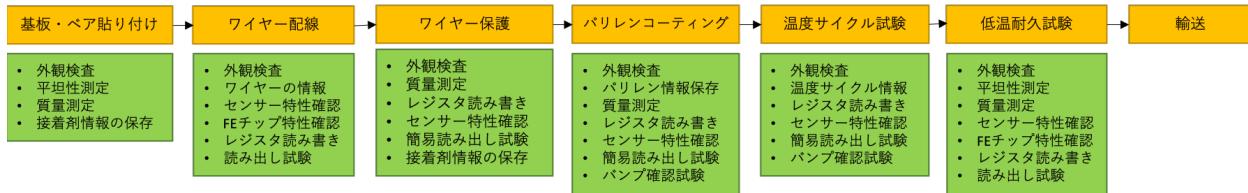


図 3.11 組み立て工程と対応する品質試験一覧。章 3.1 で述べた各組み立て工程において、複数の品質試験を行う必要がある。図は各組み立て工程と対応する品質試験一覧を示している。試験数は 30 程度存在する。

319

- 外部トリガーを用いた測定

320

3.2.7 各組み立て工程における品質試験

各組み立て工程と品質試験項目を図 3.11 に示す。図より、1 モジュールに対して行われる品質試験は 30 程度存在することが分かる。

323

3.3 検出器量産におけるデータ管理

各組み立て機関で $O(100) \sim O(1,000)$ のモジュールを作り、上述したように 1 モジュールに対して 30 程度の品質試験を行う。1,000 モジュール作る機関であれば 3,000 の試験結果が得られる。この数の品質試験結果を正確に管理することが必要となる。特に読み出し試験については、1 試験につき更に細かい項目に細分化され、項目数、結果ファイル数も多様である。これらの情報は最終的に共通のデータベースに保存する必要があり、各組み立て機関で適切に管理する必要がある。

本研究では各組み立て機関におけるモジュール情報及び品質試験のデータ管理を簡易化することを目的として、データベースシステムの構築を行った。このシステムについて 4 章で記述する。

331 第4章

332 モジュール情報及び品質試験結果管理シ 333 ステム

334 前章で述べたように、モジュール生産及び品質試験を世界中で行う。これらの情報はデータベースシ
335 テムを用いて管理することが決定していて、現在この開発を行っている。システムについては、ITk の全
336 情報を保存する中央データベースと、各組み立て機関に設置し、データ管理を行うローカルデータベース
337 である。本章ではこれらのデータベースについて説明する。また、システム開発の中で私が開発を行った
338 機能について詳細に説明する。

339 4.1 中央データベース

340 4.1.1 中央データベースの概要

341 概要

342 中央データベースは、ITk の製造に関する全ての情報の保存を目的として開発されたデータベースであ
343 る。ユニコーン大学が開発、運用を行っていて、チェコにデータベースサーバーが設けられている。ITk
344 は、ピクセル検出機とストリップ検出機にから構成される。これらを生産するにあたって、シリコンセン
345 サーやフレキシブル基板といった小さな部品から製造を行い、それらを用いたモジュールの組み立て、複
346 数モジュールを搭載したステープやリングの組み立てを経て検出器が完成する。また各組み立て段階にお
347 いて、動作確認等を目的とした品質試験を行う。これらの過程における全ての構成部品の情報、及び品質
348 試験結果を中央データベースに保存する。

349 意義

350 中央データベースを扱う一番の目的は、ITk に用いるモジュールの選別とその配置の決定に用いること
351 である。品質試験結果を解析、品質のいいモジュールを 10,000 台の中から選別、ITk に搭載する。また、
352 モジュールの配置も品質を考慮して決定する。例として以下の 2 つをあげる。

- 353 1. $|\eta|$ が小さい領域には、不良ピクセルが少なく品質の良いモジュールを搭載する.
- 354 2. 不良ピクセルの領域が固まらないようにする.

355 (もう少し何か書きたい。SUSY や higgs が低 eta に出るとか。)

356 次に中央データベースに保存された情報は、検出器運転時の参考値として扱われる。モジュールを例に

357 だすと、品質試験で読み出し試験を行った際の最適な設定値を中央データベースに保存するため、実際の
358 運転時に参照することができる。また運転前の状態における検出器の性能、運転前後での性能比較を行う
359 ことができる。

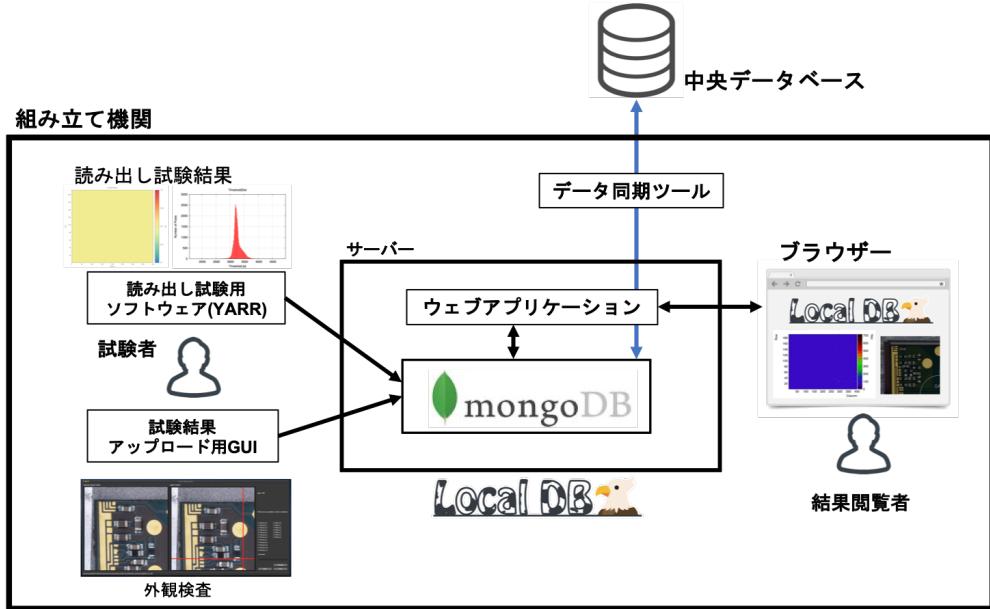


図 4.1 ローカルデータベースシステムの概要。各組み立て機関で MongoDB とローカルデータベース用ウェブアプリケーションの立ち上げを行い、独自にデータ管理をするシステムとなっている。品質試験者は図のようにいくつかのソフトウェアを用いて試験結果を MongoDB に保存する。保存された結果はウェブアプリケーションによって閲覧することができる。また結果は中央データベースに集める必要があるため、同期ツールを用いて試験結果の共有を行う。

360 4.2 ローカルデータベース

361 4.2.1 ローカルデータベースの意義と概要

362 中央データベースでは、前述したようにモジュールの情報のみならず ITk に関わるすべての情報を管
363 理する。データベースの機能としては汎用的に使えるようなものになっている。モジュールの組み立て及
364 びその品質試験に関しては 3 章で述べたように工程が複数に渡り、行う品質試験の数も多い。1 つの生産
365 現場で多いところでは数千個のモジュールを作ることになるため、データ管理が簡単にかつ円滑に進むよ
366 うになっているのが好ましい。このような理由から、生産現場での生産性、利便性に特化し、円滑な生産
367 をサポートすることを目的としたデータベースシステム（ローカルデータベース）を開発している。シス
368 テムの概要図を図 4.1 に示す。オープンソースのサービスである MongoDB[16] と、Python のウェブフ
369 レームワークである Flask[17] を使用し、開発を行なっているローカルデータベース用ウェブアプリケー
370 ションを併用することで、データ管理や中央データベースとの同期を行うシステムとなっている。

371 ローカルデータベースシステムには以下のようないちどりを持ち得る。これらの利点を持つシステムの開
372 発、実装を行うことが開発課題となっている。

- 373 • ローカルにデータベースサーバーを立てるためアクセス速度が早く、円滑にデータ管理を行うこと
374 ができる。
- 375 • モジュールの組み立て工程を管理し、生産者の適切な処理を助ける。
- 376 • モジュールに特化したデータ管理、解析を行うことで異常をいち早く検知できる。
- 377 • 試験者の情報や試験時間など、品質試験結果以外の必要な情報を正確に管理できる。

378 4.2.2 先行研究と開発課題

379 先行研究で開発された領域と、本研究で取り組んだ開発課題を以下に示す。

380 先行研究

381 先行研究 [18]においてデータベースシステムの設計と開発がなされ、図4.1におけるいくつかの項目が
382 開発された。以下に実装項目を記す。

- 383 1. 読み出し試験結果保存のためのMongoDB内部構造の設計.
384 2. 1の構造を用いてYARRからMongoDBへの読み出し試験結果アップロード機能.
385 3. 試験結果のソート及び閲覧が可能な、Flaskを用いたウェブアプリケーション.

386 モジュールの生産、品質試験に向けたデータ管理、そしてローカルデータベースが上述した利点を持つ
387 ことを達成するには以下のような開発課題が残されていた。

- 388 • 中央データベースの内部構造の実装.
389 • 中央データベースとローカルデータベース間の同期ツールの開発.
390 • ローカルデータベースにおける品質試験に特化したデータ管理と機能提供.
391 • 量産時におけるデータベース操作の流れの確立.

392 これらの開発課題に対して、本研究で取り組んだ項目を以下に挙げる。なお先行研究で開発したデータ
393 構造やアプリケーションといった項目は既に複数の機関で試験運用されていたため、本研究ではその構造
394 やソフトウェアを大きく変更せずに、拡張する形で開発を行なった。

395 中央データベースの内部構造の実装

396 モジュール及び品質試験の結果を中央データベースに登録するために、モジュールの種類や品質試験項目
397 など中央データベースにおける内部構造の実装する必要がある。本研究ではこういった量産時において
398 必要となるデータ構造の実装を行なった。

399 中央データベースとローカルデータベース間の同期ツールの開発

400 ローカルデータベースを各組み立て機関に設置し、データ管理を行う。中央データベースに必要な情報
401 が全て保存されるためにはデータベース間の同期が必要である。これを達成するために中央データベース
402 とローカルデータベースの同期を行うツール開発を行なった。

403 ローカルデータベースにおける品質試験に特化したデータ管理と機能提供

404 ローカルデータベースにおける利点の1つである「品質試験に特化したデータ管理と機能提供」を達成
405 するために以下の機能を実装した。

- 406 • 品質試験者及びシステムユーザ管理機能及びコメント付加等の各種機能.
407 • 品質試験結果の登録と組み立て工程の自動更新.
408 • 読み出し試験におけるピクセル解析ツールの開発.
409 • 読み出し試験結果の検索機能.

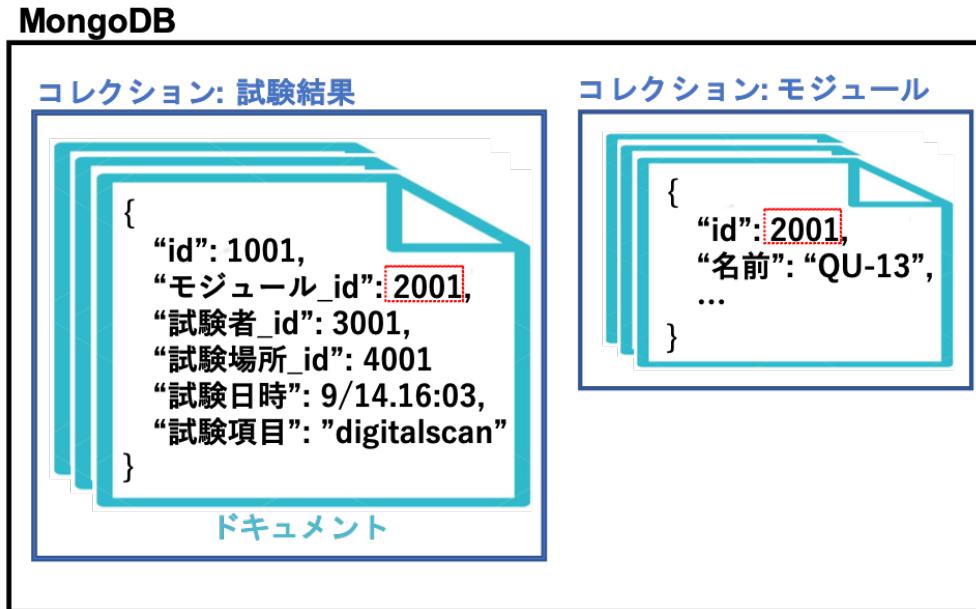


図 4.2 MongoDB の構造の例 [19]。図のように MongoDB では JSON 形式でデータを格納する。1 枚の JSON インスタンスをドキュメントと呼び、複数のドキュメントが格納されている枠組みをコレクションと呼ぶ。ドキュメントの構造及びコレクション間の関係等を決めることでデータベースの構造を定義する。ドキュメント間の紐付けは、各ドキュメント内部に ID を持つことで可能となる。図の例では試験結果のドキュメントが {“モジュール_id”:2001} の情報を持っており、これがモジュールに保存されているドキュメントの 1 枚目の ID に対応する。

410 量産時におけるデータベース操作の流れの確立

411 先行研究では品質試験の際に、登録、同期といったデータベース操作の流れが確立されていなかった。
412 本研究では上述した機能開発と併せて、品質試験におけるデータ管理の流れを確立した。

413 4.2.3 MongoDB と内部構造 [19]

414 MongoDB とは NoSQL に分類されるデータベースである。MongoDB の構造について簡単に表したもの

415 を図 4.2 に示す。一般的な SQLDB のようにテーブル形式ではなく、JSON 形式で情報を格納する。

416 情報を保持している一枚の JSON インスタンスを「ドキュメント」と呼び、「コレクション」と呼ばれる

417 枠に複数のドキュメントが格納されている。各ドキュメントは「ID」と呼ばれるハッシュ値を持ってい

418 て、異なるコレクションにおけるドキュメント間の紐付けはこの ID を用いて行う。

419 ローカルデータベースシステムにおいて、MongoDB を使用する主な利点を以下に示す。

- 420 • 各コレクションに格納するドキュメントの構造が動的であるため、開発を柔軟に行うことができる。
- 421 • JSON 形式でデータを保持するため情報取得の際の整形処理を容易であり、ウェブアプリケーションとの親和性が高い。
- 422 • データのキャッシュをメモリ上に置き処理を実行するため、高速な読み書きが可能。

425 モジュール及び品質試験に用いる主なコレクションを表 4.1 に示す。また先行研究で設計された読み出
426 し試験に関する内部構造の詳細と関係性を図 4.3 に示す。

表 4.1 品質試験に用いる主なコレクション。ローカルデータベースシステムにおいて、MongoDB 内に 2 つのデータベースを設置し、使用する。

データベース名	コレクション名	情報
localdb	component	モジュール情報、FE チップ情報
	childParentRelation	FE チップとモジュールの関係性
	QC.module.status	各モジュールに対する組み立て工程及び選択された試験結果
	QC.result	品質試験結果
	testRun	読み出し試験結果
	user	読み出し試験実施者
	institute	読み出し試験実施場所
	componentTestRun	component と testRun の関係性
localdbtools	comments	コメント情報
	QC.status	組み立て工程及び試験項目
	viewer.user	登録ユーザの情報
	viewer.query	読み出し結果キーワード、検索機能実行時に使用
	viewer.tag.docs	モジュールや試験結果に付けるタグの情報

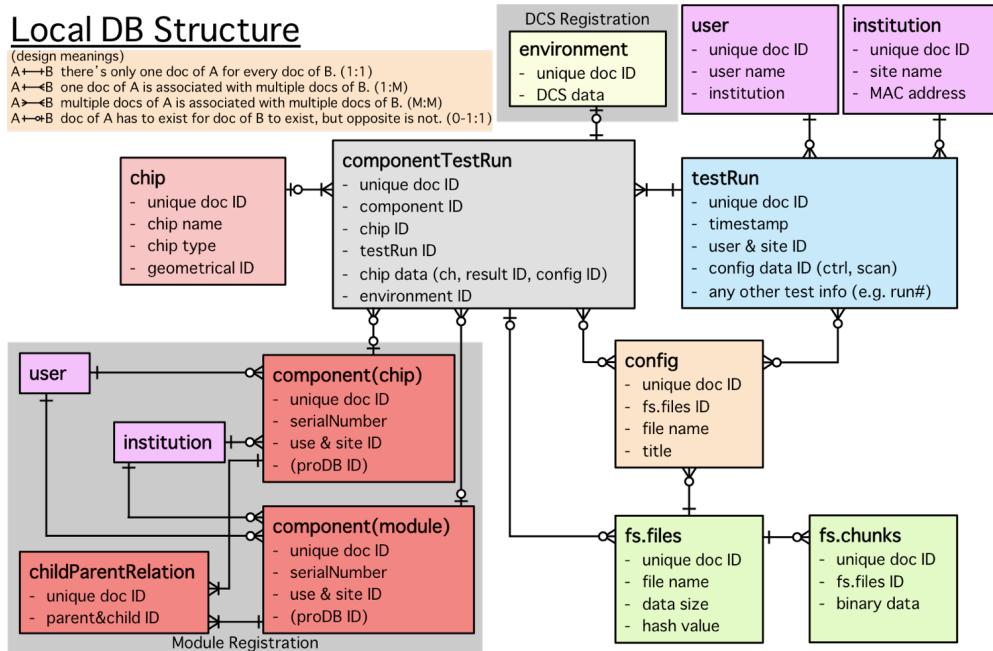


図 4.3 先行研究により設計された読み出し試験結果の MongoDB 内部構造 [18]。それぞれの四角はコレクション、直線は ID によるドキュメント間のリンクを示している。直線が十字になっている場合はドキュメント間が一対一対応であることを示し、分岐しているものは複数のドキュメントと対応していることを示す。また直線状の白丸は対応するドキュメントが存在しない場合があることを示している。

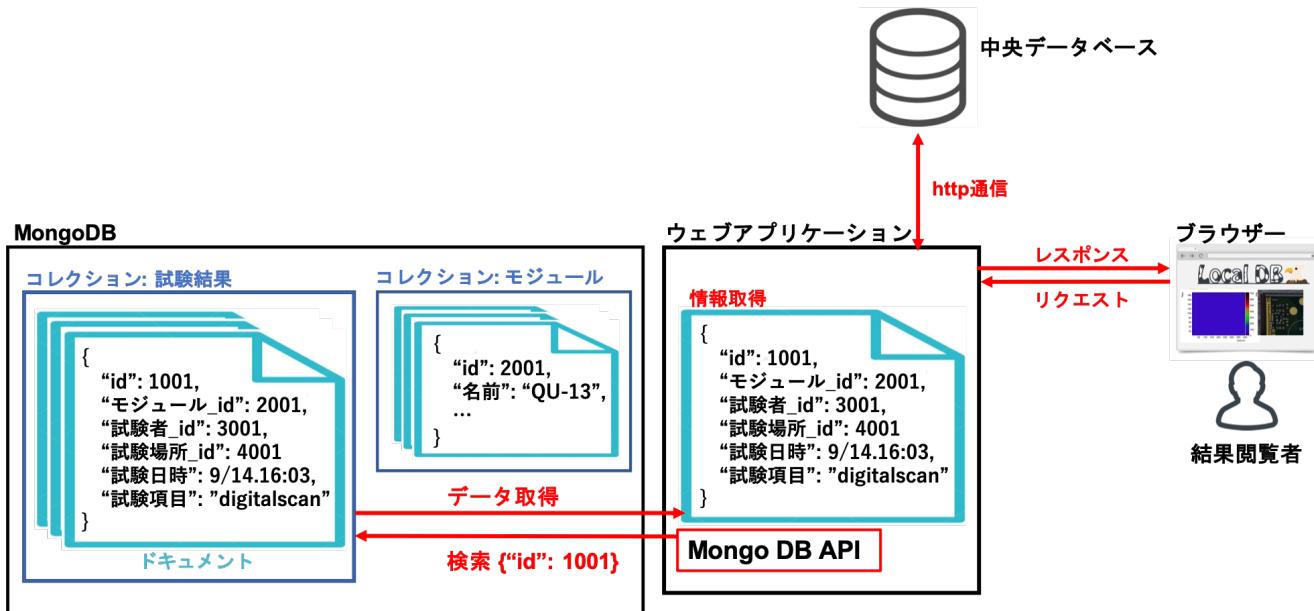


図 4.4 ウェブアプリケーション処理のイメージ。ウェブアプリケーションでは MongoDB 通信 API(PyMongo) を用いて、データベースのコレクションに検索をかけることで情報を取得する。取得した情報は整形されたのちブラウザに送信、中央データベースとの同期等の処理に用いられる。

4.2.4 ウェブアプリケーション

各組み立て機関において、試験者が品質試験結果を閲覧、管理するツールとして、ウェブアプリケーションを提供している。アプリケーション開発には、Python のウェブフレームワークである Flask を使っている。またアプリケーションにおいて MongoDB との通信に用いる API として、Python ライブラリである PyMongo[20] を用いている。ローカルデータベースとアプリケーション間の処理に特化したイメージを図 4.4 に示す。このようにアプリケーションはデータベースとブラウザ、データベース間のインターフェースとなっている。

試験結果を迅速に分かりやすく見るシステムを作り、円滑な生産の補助や異常結果の早期発見を目的としている。またデータベースの情報管理のみならず、同期ツールや、後述する試験結果解析ツールなどの外部スクリプトの実行、結果取得等、生産時における多くのデータベース操作はこのアプリケーションを用いて行う。

ウェブアプリケーションでは、現在以下の機能を使用することができる。ある品質試験の結果ページを図 4.5 に示す。

- 登録モジュール情報及び品質試験結果の閲覧、解析
- ローカルデータベースにおけるユーザ管理機能
- データベース同期実行機能

Result: 1348

Information

Component

Result

Scan

Key	Data
runNumber	1348
testType	std_digitalscan
stage	MODULEWIREBONDING
component	20UPGR00000001 20UPGFC9999999
startTime	2020/11/13 19:49:40
finishTime	2020/11/13 19:49:45
user	hokuyama
site	lazulite
targetCharge	-1
targetTot	-1
exec	-r configs/controller/specCfg.json -c db-data/connectivity.json -s configs/scans/rd53a/std_digitalscan.Json -W
stopwatch	analysis: 626 config: 37 processing: 1 scan: 2821
QC	False
environment	True
plots	L1Dist EnMask OccupancyMap
passed	True
qcTest	False
qaTest	False
summary	False

Output Data

Type	Format	Chip	Display	Download
ctrlCfg	json			
dbCfg	json			
siteCfg	json			
userCfg	json			
scanCfg	json			
beforeCfg	json	20UPGFC9999999		
afterCfg	json	20UPGFC9999999		
EnMask	json	20UPGFC9999999		
OccupancyMap	json	20UPGFC9999999		
L1Dist	json	20UPGFC9999999		

PLOT JSROOT

L1Dist plotly

20UPGFC9999999

EnMask plotly

20UPGFC9999999

OccupancyMap plotly

20UPGFC9999999

DCS plot

© 2019 ATLAS Japan ITk and [Tokyo Tech](#) with the great help of LBNL.

Released under the GNU license, please read [LICENSE.TXT](#)

Please refer to the LocalDB official document. [Link](#)

If found any problem, please contact to [Hide Oide](#), [Eunchong Kim](#), [Arisa Kubota](#), [Hiroki Okuyama](#), [Satoshi Kinoshita](#)

図 4.5 品質試験結果ページの例。図は品質試験項目であるデジタル回路読み出しの結果を表している。図の上部に試験情報や設定値、下部に結果のグラフが表示されているのを確認できる。

4.3 本研究における開発項目の詳細

443 以下は本研究で開発した項目である。

4.3.1 中央データベースの内部データ構造の実装

446 モジュール及びその品質試験に関する情報を中央データベースに情報を保存するために、情報構造の定義、実装を行う必要がある。以下の項目について、中央データベースが提供している API を用いて内部データ構造の定義を行った。

- 449 1. モジュールの種類とその構成部品 (図 4.6).
 450 2. モジュール組み立て工程と付随する品質試験 (表 4.2).

451 また項目 1 に関して、Quad モジュールに関する例を図 4.7 に示す。

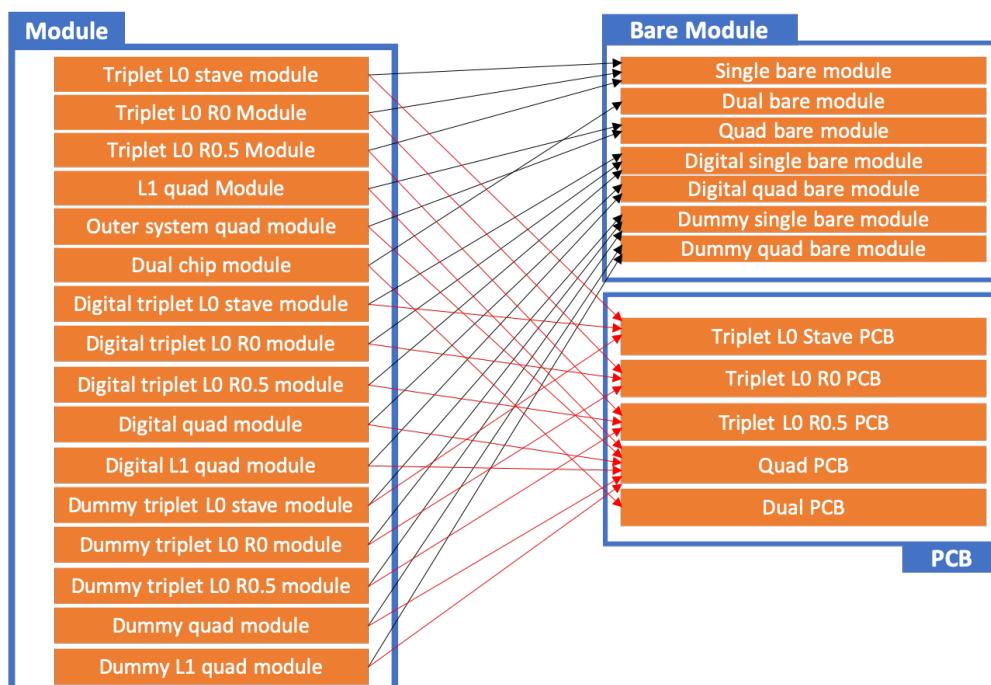


図 4.6 中央データベースにおけるモジュールの種類と構造。中央データベースにモジュールを登録するときの情報として、モジュールの種類、構成部品を図のように実装した。図の左側に実装したモジュールの種類を示しており、Triplet、Quad というように、モジュールの種類ごとに登録できるシステムとなっている。また矢印は構成部品を指しており、各モジュールは対応する Bare Module(ベアモジュール)と PCB(フレキシブル基板)を持つ。DB の中でモジュールと構成部品の紐付けも同時にを行うことができる。

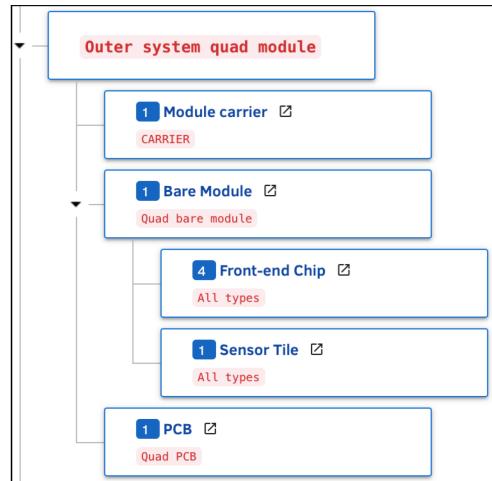


図 4.7 中央データベース内におけるモジュール構造の一例 (Quad モジュール)。例として Outer system quad module の中央データベース内の構造を示している。この種類では構成要素としてそれぞれ対応する種類の Module carrier、Bare Module、PCB を持つことがわかる。さらに Bare Module は FE chip を 4、Sensor を 1 持つことが分かり、Quad モジュールの構造が正しく実装されていることが分かる。

表 4.2 中央データベースにおける組み立て工程と付随するテスト項目。モジュールの組み立て工程及び品質試験を登録するため、表のような構造を実装した。データベース内でこの表に沿った組み立て工程の登録、更新、試験結果のアップロードができるようになった。

組み立て項目	付随する組み立て情報及び品質試験項目
1. Bare to PCB assembly	Visual Inspection Metrology Mass measurement Glue information
2. Wirebonding	Visual Inspection Wirebond information (Wirebond pull test) First power up Sensor IV SLDO VI Chip configuration Pixel failure test
3. Wirebond Protection	Visual Inspection Potting information Sensor IV Register test Readout for basic electrical
4. Parylene Coating	Visual Inspection Parylene information Mass measurement Sensor IV Register test Readout for basic electrical Bump bond quality
5. Thermal Cycling	Visual Inspection Thermal cycling info Sensor IV Register test Readout for basic electrical Bump bond quality
6. Burn-in	Visual Inspection Metrology Mass Measurement First power up Sensor IV SLDO VI Chip configuration Pixel failure test
7. Reception	

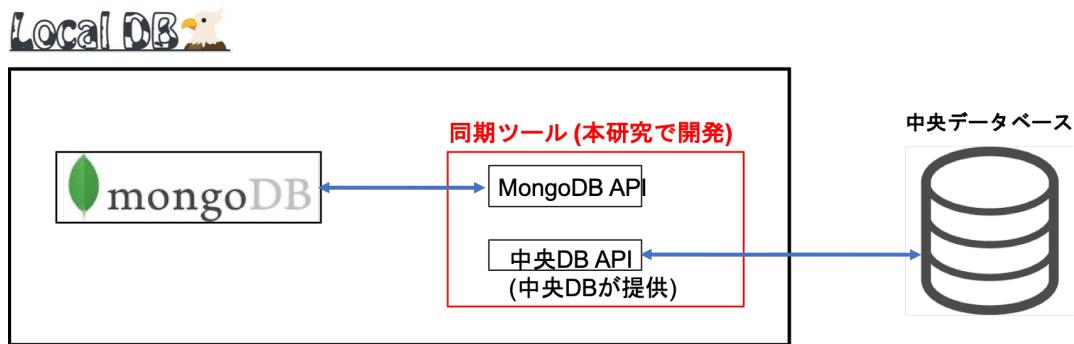


図 4.8 同期ツールのデータ通信のイメージ。本研究で開発を行っているのは図の赤線の領域に対応する同期ツールである。このツールは Python を用いて開発しており、処理の中でローカルの MongoDB と通信する API と、中央データベースが開発、提供をしている API を用いることで、2 つのデータベース間の同期を行っている。このとき、中央データベースとの通信は http 通信で行われる。

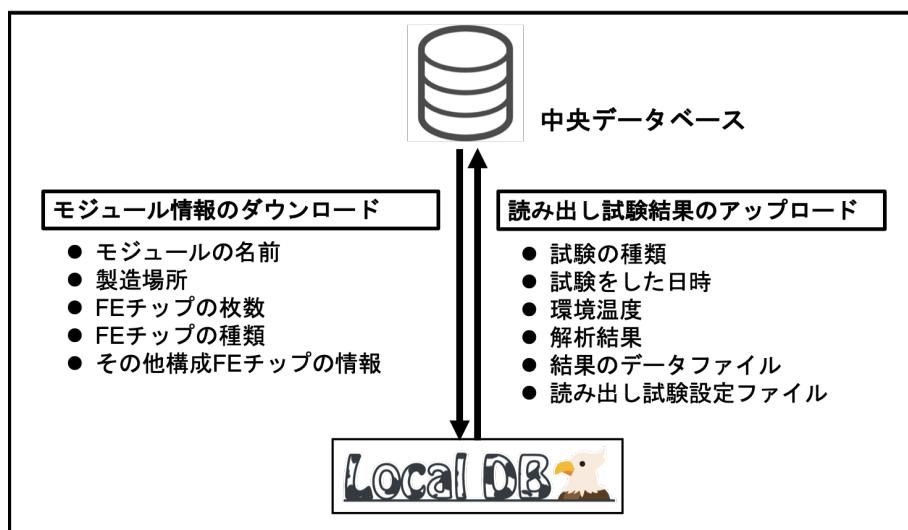


図 4.9 同期機能の概要。本研究ではモジュール情報のダウンロードと読み出し試験結果のアップロード機能を実装した。図に示している情報を同期する機能となっている。

4.3.2 データベース同期ツールの開発

モジュールや品質試験の結果のデータ共有のために、中央データベースとローカルデータベースの間で同期が行われる必要がある。これを行うツールを設計、開発を行った。

ツールの中では中央データベースが開発、提供している中央データベース通信用 API と、節??で述べたローカル MongoDB と通信する API の 2 つを用いることで情報共有を行っている。

同期ツールのデータ通信のイメージを図 4.8 に示す。

特に本研究ではツールの枠組み設計に加えて、以下の機能を実装した。

- モジュール及び構成する FE チップ情報のダウンロード機能
- 読み出し試験結果のアップロード機能

これらの機能のイメージを図 4.9 に示す。実装の詳細及び処理時間測定について 8 章で述べる。

4.3.3 ユーザ管理機能及び各種機能

異常があった際に確認することを目的として、誰が試験を行ったかを記録することが必要である。また、モジュールの登録や中央データベースとの同期など、データベースの機能使用を制限することも必要である。これらを目的として、試験者及びデータベース使用者情報の管理システムを開発、実装した。この詳細について以下に述べる。

機能概要

データベース権限の段階として、管理者、権限付きユーザ、一般ユーザの3段階を設けた。各ユーザが使うことのできる機能を表4.3に示す。

権限付きユーザの機能としてモジュール及び試験結果にコメント、タグを付ける機能を実装した。使用したときの様子を図4.10、4.11に示す。

ユーザ登録操作

表4.3において管理者と権限付ユーザの登録について説明する。

データベースシステム導入時に管理者のアカウントを作成する。コマンドプロンプト上で開発したスクリプトを用いて実行することで管理者登録がなされ、この際ユーザ名とパスワードを入力する。

権限付ユーザについて、全ての品質試験者及びデータベースユーザ機能使用者は管理者によってユーザ登録される必要がある。登録はウェブアプリケーションを用いて行い、以下の情報を入力する。

- ユーザ名
- 氏名
- 所属機関
- メールアドレス

管理者が登録を完了すると、登録されたメールアドレスに登録完了メールと仮パスワードが届く。このメールに従い、ウェブアプリケーション上でユーザがパスワード登録を完了する。

このようにメール機能を用いることでパスワード漏洩の防止、管理者操作の削減を目的としている。

表 4.3 ローカルデータベースユーザ権限及び使用機能一覧。ローカルデータシステムにおけるユーザとして、管理者、権限付きユーザ、一般ユーザの3つを設けた。全てのユーザがウェブアプリケーションの閲覧をすることができる。管理者、権限付きユーザにはデータベース読み書き権限とウェブアプリケーションログイン権限が与えられ、試験結果のアップロード、アプリケーション上のユーザ機能の実行ができる。また管理者は権限付きユーザを登録することができる。

ユーザ	付加される権限	使用できる機能
管理者	ユーザ管理権限 データベース読み書き権限 ウェブアプリケーションログイン権限	権限付きユーザ登録機能
権限付ユーザ	データベース読み書き権限 ウェブアプリケーションログイン権限	試験結果のアップロード 中央データベースとのデータ同期機能 その他ウェブアプリケーションの機能（コメント、タグ）
一般ユーザ		モジュール情報及び試験結果の閲覧

図 4.10 ウェブアプリケーションにおけるコメント機能。権限付きユーザ及び管理者はモジュールや試験結果に対してコメントをすることができる。図のようにページの右側にコメント欄があり、コメントをテキスト形式で記述することができる。

図 4.11 ウェブアプリケーションにおけるタグ機能。権限付きユーザ及び管理者はモジュールや試験結果に対してタグをつけることができる。図は試験結果の一覧ページであり、図の表において一番右の列がつけられたタグを示しており、図では anomaly や good といったタグが付けられていることが分かる。

485 機能の仕組み

486 ユーザ登録の際には内部で以下の2つの処理が行われるように実装した。

- 487 1. MongoDBアカウントの作成、読み書き権限の付与
 488 2. ウェブアプリケーションで用いるユーザ情報ドキュメントの作成

489 1の処理を行う理由は、登録ユーザが試験結果をMongoDBにアップロードできるようにするためにある。2の情報は、ウェブアプリケーション内でのログイン判断、ユーザの情報保持に使う。この情報は表4.1のviewer.userに保存される。2つの処理について、実際に保存されるドキュメントの例をリスト4.1、4.2以下に示す。

ソースコード4.1 MongoDBアカウント情報を持つドキュメントの例。リスト中の”roles”より、localdbとlocaldbtoolsの読み書き権限が付加されていることが分かる。

```
493 1 {
494 2   "_id" : "localdb.hokuyama",
495 3   "userId" : UUID("fee321eb-83b8-434a-a4a0-fff638b5db36"),
496 4   "user" : "hokuyama",
497 5   "db" : "localdb",
498 6   "credentials" : {
499 7     ...
500 8   },
501 9   "roles" : [
502 10    {
503 11      "role" : "readWrite",
504 12      "db" : "localdb"
505 13    },
506 14    {
507 15      "role" : "readWrite",
508 16      "db" : "localdbtools"
509 17    }
510 18  ]
511 19 }
```

ソースコード4.2 ウェブアプリケーションで扱うユーザ情報を持つドキュメントの例。リスト4.1で示したものとは別に、ウェブアプリケーション内でユーザ情報を扱うためにこのドキュメントを保持する必要がある。ウェブにおいてログインはこのドキュメントの存在確認をもってなされる。パスワードはhash化して保存している。

```
514 1 {
515 2   "_id" : ObjectId("5f0bbe84ef87af2628865de7"),
516 3   "sys" : {
517 4     "rev" : 0,
518 5     "cts" : ISODate("2020-07-13T10:53:07.943Z"),
519 6     "mts" : ISODate("2020-07-13T10:53:07.943Z")
520 7   },
521 8   "username" : "hokuyama",
522 9   "name" : "Hiroki\Okuyama",
523 10  "auth" : "readWrite",
524 11  "institution" : "Tokyo\Institute\of\Technology",
525 12  "Email" : "okuyama@hep.phys.titech.ac.jp",
526 13  "password" : "5f4dcc3b5aa765d61d8327deb882cf99"
527 14 }
```

530 4.3.4 品質試験結果の登録と組み立て工程の自動更新

531 ローカルデータベースへアップロードした品質試験結果の中から、本結果として中央データベースへ
 532 アップロードする結果を選択する機能を開発した。品質試験は各モジュール、各組み立て工程に対して行
 533 うものであるため、結果選択も同様に工程毎に行うことを想定している。結果選択後、データベースにお
 534 ける組み立て工程の情報は次のものへ自動的に更新する機能となっている。

535 概要

536 あるモジュール、組み立て工程に対して結果を選択する様子を図 4.12 に示す。組み立て工程も自動更
 537 新されていることがわかる。

538 仕組み

539 リスト 4.3、4.4 のようなドキュメントを作成、保存する。リスト 4.3 は全てのモジュールに対して共通
 540 のドキュメントであり、組み立て工程と各工程における品質試験項目を記録する。これらの情報は中央
 541 データベースより取得される。この情報を参照することでローカルデータベース内部での組み立て工程の
 542 管理が可能となっている。

543 リスト 4.4 は各モジュールに対して 1 つ存在し、以下のような情報を保持する。

- 544 • 現在工程
- 545 • 各工程における品質試験結果の ID

ソースコード 4.3 組み立て工程及び品質試験一覧情報ドキュメント。このようなドキュメントを作成、保持しておくことで組み立て工程及び品質試験の情報を扱う。ローカルデータベース内に 1 つこのドキュメントを保持し、品質試験結果選択、組み立て工程の更新時にこのドキュメントを参照する。このドキュメントは中央データベースよりデータ取得して作成する。

```

546 1 {
547 2   "_id" : ObjectId("5fc89aa232d56b29091fd64d"),
548 3   "sys" : {
549 4     "mts" : ISODate("2020-12-03T07:58:26.310Z"),
550 5     "cts" : ISODate("2020-12-03T07:58:26.310Z"),
551 6     "rev" : 0
552 7   },
553 8   "dbVersion" : 1.01,
554 9   "proddbVersion" : 1.01,
555 10  "stage_flow" : [
556 11    "MODULETOPCB",
557 12    "MODULEWIREBONDING",
558 13    "MODULEWIREBONDPROTECTION",
559 14    "MODULEPARYLENECOATING",
560 15    "MODULETHERMALCYCLING",
561 16    "MODULEBURNIN",
562 17    "MODULERECEPTION"
563 18  ],
564 19  "stage_test" : {
565 20    "MODULETOPCB" : [
566 21      "OPTICAL",
567 22      "GLUE_MODULE_FLEX_ATTACH",
568 23      "MASS",
569 24      "METROLOGY"
570 25    ],
571 26    "MODULEWIREBONDING" : [
572 27      "WIREBONDING",
573 28      "OPTICAL",
574 29      "SENSOR_IV",
575 30      "PIXEL_FAILURE_TEST",
576 31      "SLDO_VI",
577 32      "WIREBOND",
578 33    ]
579 34  }
580 35}
581 36
582 37
583 38
584 39
585 40
586 41
587 42
588 43
589 44
590 45
591 46
592 47
593 48
594 49
595 50
596 51
597 52
598 53
599 54
600 55
601 56
602 57
603 58
604 59
605 60
606 61
607 62
608 63
609 64
610 65
611 66
612 67
613 68
614 69
615 70
616 71
617 72
618 73
619 74
620 75
621 76
622 77
623 78
624 79
625 80
626 81
627 82
628 83
629 84
630 85
631 86
632 87
633 88
634 89
635 90
636 91
637 92
638 93
639 94
640 95
641 96
642 97
643 98
644 99
645 100
646 101
647 102
648 103
649 104
650 105
651 106
652 107
653 108
654 109
655 110
656 111
657 112
658 113
659 114
660 115
661 116
662 117
663 118
664 119
665 120
666 121
667 122
668 123
669 124
670 125
671 126
672 127
673 128
674 129
675 130
676 131
677 132
678 133
679 134
680 135
681 136
682 137
683 138
684 139
685 140
686 141
687 142
688 143
689 144
690 145
691 146
692 147
693 148
694 149
695 150
696 151
697 152
698 153
699 154
700 155
701 156
702 157
703 158
704 159
705 160
706 161
707 162
708 163
709 164
710 165
711 166
712 167
713 168
714 169
715 170
716 171
717 172
718 173
719 174
720 175
721 176
722 177
723 178
724 179
725 180
726 181
727 182
728 183
729 184
730 185
731 186
732 187
733 188
734 189
735 190
736 191
737 192
738 193
739 194
740 195
741 196
742 197
743 198
744 199
745 200
746 201
747 202
748 203
749 204
750 205
751 206
752 207
753 208
754 209
755 210
756 211
757 212
758 213
759 214
760 215
761 216
762 217
763 218
764 219
765 220
766 221
767 222
768 223
769 224
770 225
771 226
772 227
773 228
774 229
775 230
776 231
777 232
778 233
779 234
780 235
781 236
782 237
783 238
784 239
785 240
786 241
787 242
788 243
789 244
790 245
791 246
792 247
793 248
794 249
795 250
796 251
797 252
798 253
799 254
800 255
801 256
802 257
803 258
804 259
805 260
806 261
807 262
808 263
809 264
810 265
811 266
812 267
813 268
814 269
815 270
816 271
817 272
818 273
819 274
820 275
821 276
822 277
823 278
824 279
825 280
826 281
827 282
828 283
829 284
830 285
831 286
832 287
833 288
834 289
835 290
836 291
837 292
838 293
839 294
840 295
841 296
842 297
843 298
844 299
845 300
846 301
847 302
848 303
849 304
850 305
851 306
852 307
853 308
854 309
855 310
856 311
857 312
858 313
859 314
860 315
861 316
862 317
863 318
864 319
865 320
866 321
867 322
868 323
869 324
870 325
871 326
872 327
873 328
874 329
875 330
876 331
877 332
878 333
879 334
880 335
881 336
882 337
883 338
884 339
885 340
886 341
887 342
888 343
889 344
890 345
891 346
892 347
893 348
894 349
895 350
896 351
897 352
898 353
899 354
900 355
901 356
902 357
903 358
904 359
905 360
906 361
907 362
908 363
909 364
910 365
911 366
912 367
913 368
914 369
915 370
916 371
917 372
918 373
919 374
920 375
921 376
922 377
923 378
924 379
925 380
926 381
927 382
928 383
929 384
930 385
931 386
932 387
933 388
934 389
935 390
936 391
937 392
938 393
939 394
940 395
941 396
942 397
943 398
944 399
945 400
946 401
947 402
948 403
949 404
950 405
951 406
952 407
953 408
954 409
955 410
956 411
957 412
958 413
959 414
960 415
961 416
962 417
963 418
964 419
965 420
966 421
967 422
968 423
969 424
970 425
971 426
972 427
973 428
974 429
975 430
976 431
977 432
978 433
979 434
980 435
981 436
982 437
983 438
984 439
985 440
986 441
987 442
988 443
989 444
990 445
991 446
992 447
993 448
994 449
995 450
996 451
997 452
998 453
999 454
1000 455
1001 456
1002 457
1003 458
1004 459
1005 460
1006 461
1007 462
1008 463
1009 464
1010 465
1011 466
1012 467
1013 468
1014 469
1015 470
1016 471
1017 472
1018 473
1019 474
1020 475
1021 476
1022 477
1023 478
1024 479
1025 480
1026 481
1027 482
1028 483
1029 484
1030 485
1031 486
1032 487
1033 488
1034 489
1035 490
1036 491
1037 492
1038 493
1039 494
1040 495
1041 496
1042 497
1043 498
1044 499
1045 500
1046 501
1047 502
1048 503
1049 504
1050 505
1051 506
1052 507
1053 508
1054 509
1055 510
1056 511
1057 512
1058 513
1059 514
1060 515
1061 516
1062 517
1063 518
1064 519
1065 520
1066 521
1067 522
1068 523
1069 524
1070 525
1071 526
1072 527
1073 528
1074 529
1075 530
1076 531
1077 532
1078 533
1079 534
1080 535
1081 536
1082 537
1083 538
1084 539
1085 540
1086 541
1087 542
1088 543
1089 544
1090 545
1091 546
1092 547
1093 548
1094 549
1095 550
1096 551
1097 552
1098 553
1099 554
1100 555
1101 556
1102 557
1103 558
1104 559
1105 560
1106 561
1107 562
1108 563
1109 564
1110 565
1111 566
1112 567
1113 568
1114 569
1115 570
1116 571
1117 572
1118 573
1119 574
1120 575
1121 576
1122 577
1123 578
1124 579
1125 580
1126 581
1127 582
1128 583
1129 584
1130 585
1131 586
1132 587
1133 588
1134 589
1135 590
1136 591
1137 592
1138 593
1139 594
1140 595
1141 596
1142 597
1143 598
1144 599
1145 600
1146 601
1147 602
1148 603
1149 604
1150 605
1151 606
1152 607
1153 608
1154 609
1155 610
1156 611
1157 612
1158 613
1159 614
1160 615
1161 616
1162 617
1163 618
1164 619
1165 620
1166 621
1167 622
1168 623
1169 624
1170 625
1171 626
1172 627
1173 628
1174 629
1175 630
1176 631
1177 632
1178 633
1179 634
1180 635
1181 636
1182 637
1183 638
1184 639
1185 640
1186 641
1187 642
1188 643
1189 644
1190 645
1191 646
1192 647
1193 648
1194 649
1195 650
1196 651
1197 652
1198 653
1199 654
1200 655
1201 656
1202 657
1203 658
1204 659
1205 660
1206 661
1207 662
1208 663
1209 664
1210 665
1211 666
1212 667
1213 668
1214 669
1215 670
1216 671
1217 672
1218 673
1219 674
1220 675
1221 676
1222 677
1223 678
1224 679
1225 680
1226 681
1227 682
1228 683
1229 684
1230 685
1231 686
1232 687
1233 688
1234 689
1235 690
1236 691
1237 692
1238 693
1239 694
1240 695
1241 696
1242 697
1243 698
1244 699
1245 700
1246 701
1247 702
1248 703
1249 704
1250 705
1251 706
1252 707
1253 708
1254 709
1255 710
1256 711
1257 712
1258 713
1259 714
1260 715
1261 716
1262 717
1263 718
1264 719
1265 720
1266 721
1267 722
1268 723
1269 724
1270 725
1271 726
1272 727
1273 728
1274 729
1275 730
1276 731
1277 732
1278 733
1279 734
1280 735
1281 736
1282 737
1283 738
1284 739
1285 740
1286 741
1287 742
1288 743
1289 744
1290 745
1291 746
1292 747
1293 748
1294 749
1295 750
1296 751
1297 752
1298 753
1299 754
1300 755
1301 756
1302 757
1303 758
1304 759
1305 760
1306 761
1307 762
1308 763
1309 764
1310 765
1311 766
1312 767
1313 768
1314 769
1315 770
1316 771
1317 772
1318 773
1319 774
1320 775
1321 776
1322 777
1323 778
1324 779
1325 780
1326 781
1327 782
1328 783
1329 784
1330 785
1331 786
1332 787
1333 788
1334 789
1335 790
1336 791
1337 792
1338 793
1339 794
1340 795
1341 796
1342 797
1343 798
1344 799
1345 800
1346 801
1347 802
1348 803
1349 804
1350 805
1351 806
1352 807
1353 808
1354 809
1355 810
1356 811
1357 812
1358 813
1359 814
1360 815
1361 816
1362 817
1363 818
1364 819
1365 820
1366 821
1367 822
1368 823
1369 824
1370 825
1371 826
1372 827
1373 828
1374 829
1375 830
1376 831
1377 832
1378 833
1379 834
1380 835
1381 836
1382 837
1383 838
1384 839
1385 840
1386 841
1387 842
1388 843
1389 844
1390 845
1391 846
1392 847
1393 848
1394 849
1395 850
1396 851
1397 852
1398 853
1399 854
1400 855
1401 856
1402 857
1403 858
1404 859
1405 860
1406 861
1407 862
1408 863
1409 864
1410 865
1411 866
1412 867
1413 868
1414 869
1415 870
1416 871
1417 872
1418 873
1419 874
1420 875
1421 876
1422 877
1423 878
1424 879
1425 880
1426 881
1427 882
1428 883
1429 884
1430 885
1431 886
1432 887
1433 888
1434 889
1435 890
1436 891
1437 892
1438 893
1439 894
1440 895
1441 896
1442 897
1443 898
1444 899
1445 900
1446 901
1447 902
1448 903
1449 904
1450 905
1451 906
1452 907
1453 908
1454 909
1455 910
1456 911
1457 912
1458 913
1459 914
1460 915
1461 916
1462 917
1463 918
1464 919
1465 920
1466 921
1467 922
1468 923
1469 924
1470 925
1471 926
1472 927
1473 928
1474 929
1475 930
1476 931
1477 932
1478 933
1479 934
1480 935
1481 936
1482 937
1483 938
1484 939
1485 940
1486 941
1487 942
1488 943
1489 944
1490 945
1491 946
1492 947
1493 948
1494 949
1495 950
1496 951
1497 952
1498 953
1499 954
1500 955
1501 956
1502 957
1503 958
1504 959
1505 960
1506 961
1507 962
1508 963
1509 964
1510 965
1511 966
1512 967
1513 968
1514 969
1515 970
1516 971
1517 972
1518 973
1519 974
1520 975
1521 976
1522 977
1523 978
1524 979
1525 980
1526 981
1527 982
1528 983
1529 984
1530 985
1531 986
1532 987
1533 988
1534 989
1535 990
1536 991
1537 992
1538 993
1539 994
1540 995
1541 996
1542 997
1543 998
1544 999
1545 1000
1546 1001
1547 1002
1548 1003
1549 1004
1550 1005
1551 1006
1552 1007
1553 1008
1554 1009
1555 1010
1556 1011
1557 1012
1558 1013
1559 1014
1560 1015
1561 1016
1562 1017
1563 1018
1564 1019
1565 1020
1566 1021
1567 1022
1568 1023
1569 1024
1570 1025
1571 1026
1572 1027
1573 1028
1574 1029
1575 1030
1576 1031
1577 1032
1578 1033
1579 1034
1580 1035
1581 1036
1582 1037
1583 1038
1584 1039
1585 1040
1586 1041
1587 1042
1588 1043
1589 1044
1590 1045
1591 1046
1592 1047
1593 1048
1594 1049
1595 1050
1596 1051
1597 1052
1598 1053
1599 1054
1600 1055
1601 1056
1602 1057
1603 1058
1604 1059
1605 1060
1606 1061
1607 1062
1608 1063
1609 1064
1610 1065
1611 1066
1612 1067
1613 1068
1614 1069
1615 1070
1616 1071
1617 1072
1618 1073
1619 1074
1620 1075
1621 1076
1622 1077
1623 1078
1624 1079
1625 1080
1626 1081
1627 1082
1628 1083
1629 1084
1630 1085
1631 1086
1632 1087
1633 1088
1634 1089
1635 1090
1636 1091
1637 1092
1638 1093
1639 1094
1640 1095
1641 1096
1642 1097
1643 1098
1644 1099
1645 1100
1646 1101
1647 1102
1648 1103
1649 1104
1650 1105
1651 1106
1652 1107
1653 1108
1654 1109
1655 1110
1656 1111
1657 1112
1658 1113
1659 1114
1660 1115
1661 1116
1662 1117
1663 1118
1664 1119
1665 1120
1666 1121
1667 1122
1668 1123
1669 1124
1670 1125
1671 1126
1672 1127
1673 1128
1674 1129
1675 1130
1676 1131
1677 1132
1678 1133
1679 1134
1680 1135
1681 1136
1682 1137
1683 1138
1684 1139
1685 1140
1686 1141
1687 1142
1688 1143
1689 1144
1690 1145
1691 1146
1692 1147
1693 1148
1694 1149
1695 1150
1696 1151
1697 1152
1698 1153
1
```

```

579 33     "CHIP_CONFIGURATION"
580 34   ],
581 35   "MODULEWIREBONDPROTECTION" : [
582 36     "OPTICAL",
583 37     "POTTING",
584 38     "MASS",
585 39     "READOUT_IN_BASIC_ELECTRICAL_TEST",
586 40     "SENSOR_IV",
587 41     "REGISTER_TEST"
588 42   ],
589 43   ...
590 44 },
591 45 ...
592 46 }

```

ソースコード 4.4 モジュールの組み立て工程及び品質試験結果管理のためのドキュメント例。各モジュールにおいて現在の組み立て工程及び選択された品質試験結果がこのドキュメントに保存される。ドキュメント内の”currentStage”に現工程を保持する。また選択した試験結果の ID を”QC_results”に各組み立て工程ごとに持つようになっている。

```

594 1 {
595 2   "_id" : ObjectId("5fc4be4c12a45922a91b0e75"),
596 3   "sys" : {
597 4     "mts" : ISODate("2020-11-30T09:41:32.411Z"),
598 5     "cts" : ISODate("2020-11-30T09:41:32.411Z"),
599 6     "rev" : 0
600 7   },
601 8   "dbVersion" : 1.01,
602 9   "proddbVersion" : 1.01,
603 10  "component" : "5fa79114e615fa000a1a5976",
604 11  "currentStage" : "MODULEWIREBONDPROTECTION",
605 12  "latestSyncedStage" : "MODULEWIREBONDING",
606 13  "status" : "created",
607 14  "rework_stage" : [],
608 15  "QC_results" : {
609 16    "MODULETOPCB" : {
610 17      "OPTICAL" : "5fc4c2cfb6c93d451e2c9ac1",
611 18      "GLUE_MODULE_FLEX_ATTACH" : "-1",
612 19      "MASS" : "5fc4c2da27766dc6e89c024f",
613 20      "METROLOGY" : "5fc4c2eaf1f19d9cb5859f00"
614 21    },
615 22    "MODULEWIREBONDING" : {
616 23      "WIREBONDING" : "-1",
617 24      "OPTICAL" : "5fc4c4c8b7d0c86912b4958f",
618 25      "SENSOR_IV" : "5fc4c59e9e283a57ccaa1088",
619 26      "PIXEL_FAILURE_TEST" : "5fca342f6e9f1f5eafedfb92",
620 27      "SLDO_VI" : "-1",
621 28      "WIREBOND" : "-1",
622 29      "CHIP_CONFIGURATION" : "-1"
623 30    },
624 31    "MODULEWIREBONDPROTECTION" : {
625 32      "OPTICAL" : "-1",
626 33      "POTTING" : "-1",
627 34      "MASS" : "-1",
628 35      "READOUT_IN_BASIC_ELECTRICAL_TEST" : "-1",
629 36      "SENSOR_IV" : "-1",
630 37      "REGISTER_TEST" : "-1"
631 38    },
632 39    ...
633 40  }
634 41 }

```

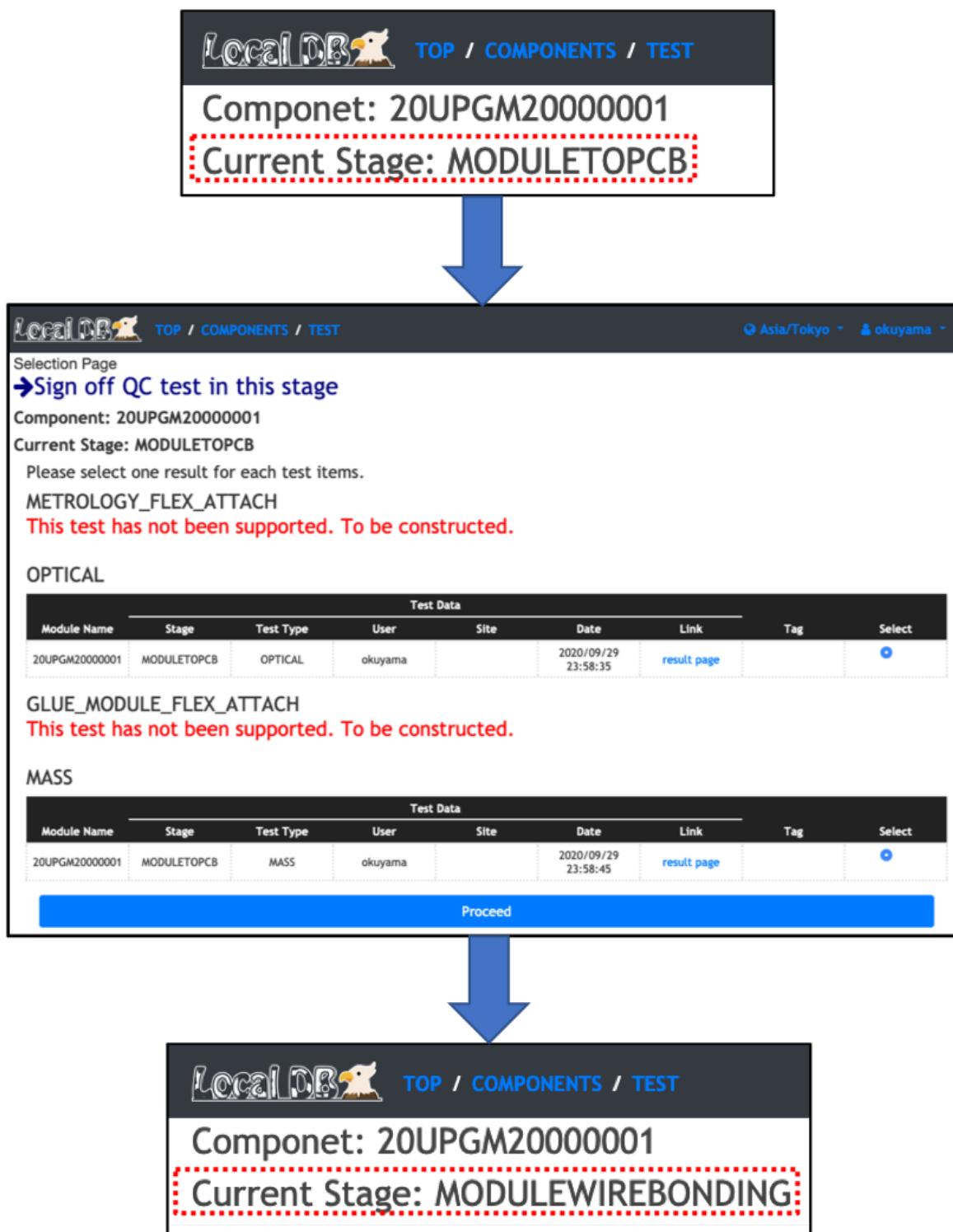


図 4.12 結果選択画面及び組み立て工程表示の例。図の上部で組み立て工程が”MODULETOPCB”である。この段階において結果を選択する処理を行うとローカルデータベース内で選択された結果にタグ付けがなされる。図の中部においてその処理を行なっており、この図では”OPTICAL”と”MASS”的結果を選択している。選択した結果は中央データベースと同期される。また結果選択後は組み立て工程が自動的に更新される。図の下部では”MODULEWIREBONDING”になっていることが分かる。

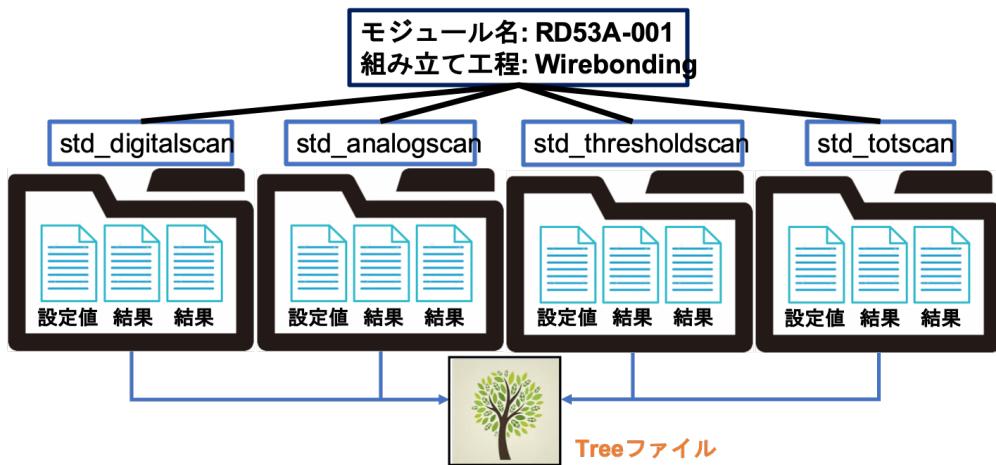


図 4.13 ピクセル解析ツールにおけるファイル統合処理のイメージ。YARR の出力ファイル及びディレクトリは std_digitalscan や std_thresholdscan というように読み出し項目ごとである。ピクセル解析ツールでは、図のようにあるモジュールに関する結果ファイルを統合し、ピクセルごとに行う解析処理を簡易化する狙いがある。

637 4.3.5 読み出し試験結果におけるピクセル解析ツール

638 節 3.2.6 で述べたように、読み出し試験ではピクセル解析を行う。これを円滑に行うために、ピクセル
639 解析ツールを開発した。また開発した解析ツールをローカルデータベースシステムに組み込んだ。この
640 ツールについての詳細を以下に示す。

641 概要

642 YARR で読み出し試験を行った場合、結果ファイル及びディレクトリは各試験項目ごとにわかれて生
643 成される。また各結果ファイルにはモジュール上の全ピクセル結果が JSON の形で保存されている。

644 一方、ピクセル解析において、いくつかの試験結果を統一的に扱い、各ピクセルごとに解析を行う必要
645 がある。そこで、開発した解析ツールでは複数の結果ファイルを 1 つに統合し、ピクセルごとの解析処理
646 を単純化する役割を担っている。開発には Python と C++ を用いた。また CERN が提供している解析
647 フレームワークである ROOT[21] を使用し、いくつかの試験データの統一ファイルとして、ROOT 内部
648 機能である Tree を使用した。このファイル統合処理のイメージを図 4.13 に示す。

649 実際に作った Tree ファイルと、データ構造のイメージを図 4.14 に示す。

650 ツールの内部構造と処理の流れ

651 開発したツールは、主に以下で説明する 3 つの実行ファイルで構成される。それぞれの役割について説
652 明する。

653 getData.py (Python)

654 データベースから対象となるデータファイルを取得、キャッシュファイルとしてサーバー上の一時
655 ディレクトリに保存。

656 makeTree (C++)

657 getData.py を用いて生成されたキャッシュファイルを読み込み、Tree ファイルを作成。



図 4.14 Tree ファイルとそのデータ保持。実際にこのツールを用いて作った Tree ファイルの内部構造の様子(左)とそのデータ保持のイメージ(右図)を示す。Tree ファイルでは、右図のように 1 つの表に試験結果をまとめている。各行が std_digitalscan といった各読み出し項目に対応し、各列が 1 ピクセルに対応する。モジュール上の行列 (Row, Col) の番号を表の上部に持っておくことで、モジュール上におけるピクセルの位置情報を保持する。

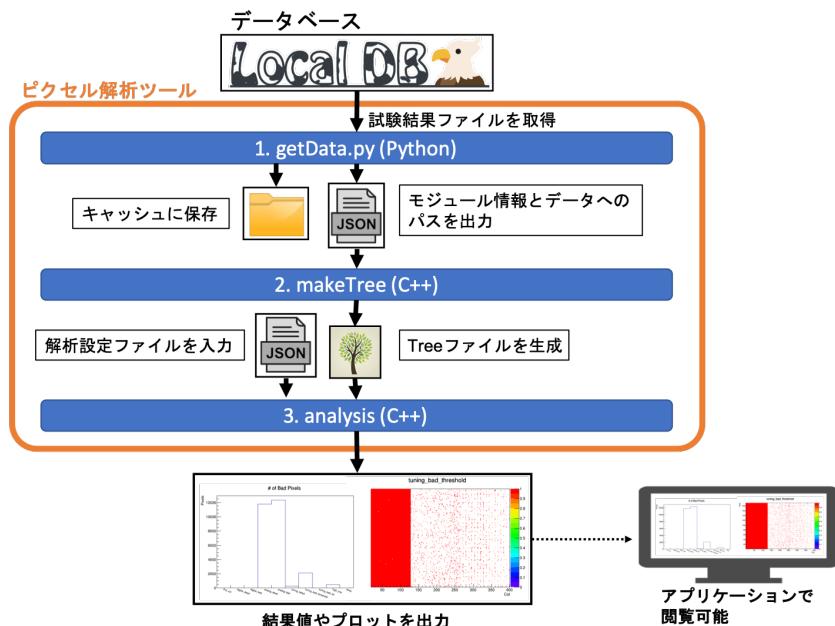
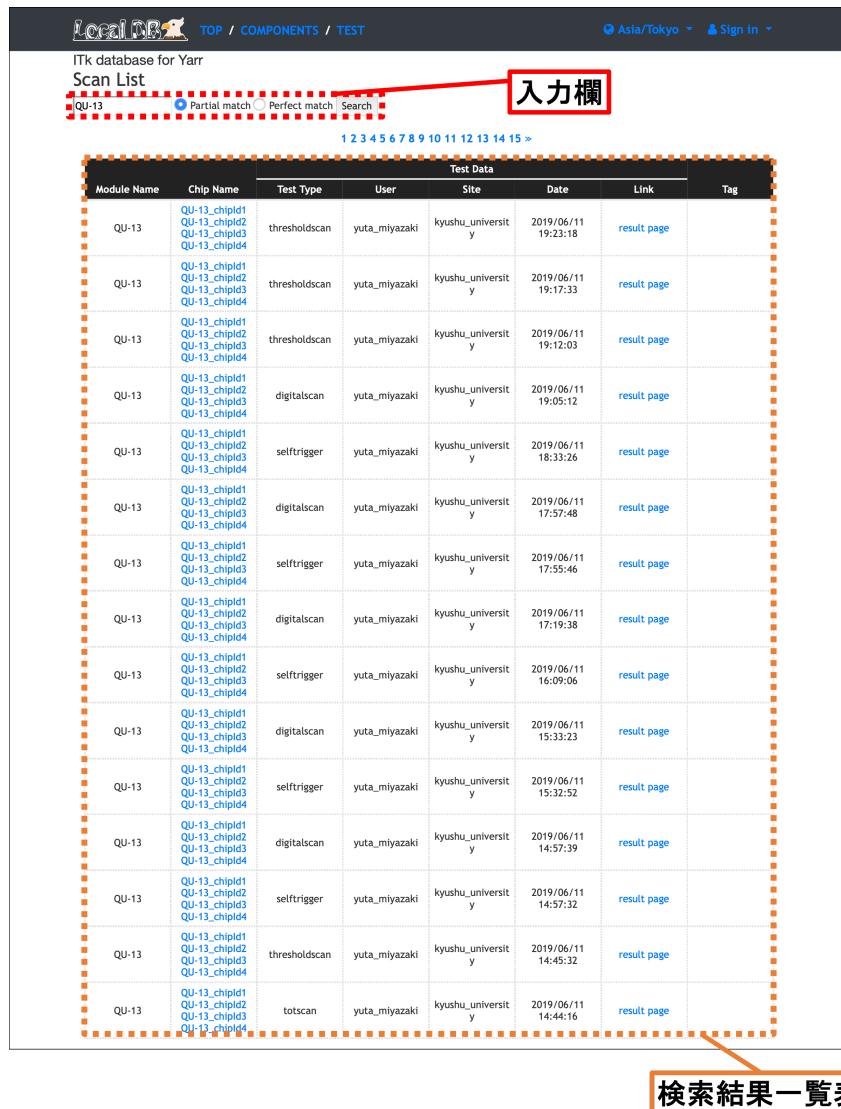


図 4.15 ピクセル解析ツールの処理の流れ。ピクセル解析ツールは、図のように 3 つの実行ファイルにより構成される。getData.py(Python) にてデータベースから結果の取得を行い、makeTree(C++) にて Tree ファイルの作成、analysis(C++) にてピクセル解析及び結果のプロットが出力される。

658 analysis (C++)

659 作成した Tree ファイルを読み込みピクセル解析を実行、結果値やプロットを出力。

660 処理の流れのイメージを図 4.15 に示す。データベースとの通信に関しては MongoDB や現システムと
661 の親和性を考慮し、Python を使用した。Tree ファイル作成やその後の解析処理のスクリプトは、ROOT
662 を使用する観点から C++ を使用した。またピクセル解析以外の解析に対しても適応可能とするため、
663 Tree 作成部と解析処理部のファイルは分割した。



The screenshot shows a web-based database application titled "LocalDB" with a "TEST" component selected. At the top, there is a search bar with the placeholder "Search" and a dropdown menu for "Partial match" or "Perfect match". Below the search bar is a table titled "Test Data" with columns: Module Name, Chip Name, Test Type, User, Site, Date, Link, and Tag. The table lists 15 rows of data, all corresponding to "Module Name": QU-13 and "Chip Name": QU-13_chipd1, QU-13_chipd2, QU-13_chipd3, QU-13_chipd4. The "Test Type" column includes "thresholdscan", "selftrigger", and "digitalscan". The "User" column is consistently "yuta_miyazaki". The "Site" column is "kyushu_university". The "Date" column shows various dates from June 11, 2019, to June 14, 2019. The "Link" column contains blue hyperlinks labeled "result page".

図 4.16 ウェブアプリケーションにおける検索機能の様子。図は検索結果一覧表示のページである。図の上部に入力欄があり（赤破線）、ここにキーワードを入力し検索を実行する。図の例では”QU-13”と入力しており、検索結果にはモジュール名 QU-13 の試験結果が一覧表示されていることが分かる。

4.3.6 読み出し試験結果の検索機能

登録モジュールや品質試験結果の一覧ページに検索機能を実装した。確認したいモジュール情報や試験結果を迅速に取得し、閲覧することを目的としている。検索機能を使用している様子を図 4.16 に示す。キーワードを入力し、検索することができる仕組みとなっていて、一般的なウェブページの検索エンジンのように扱うことができる。生産に向けて、検索にかかる処理時間測定を行った。検索機能実装方法の詳細と処理時間についての詳細は、6 章で述べる。現在は单一キーワード検索の他に、以下の機能を実装している。

- 完全一致、部分一致検索
- AND、OR 検索

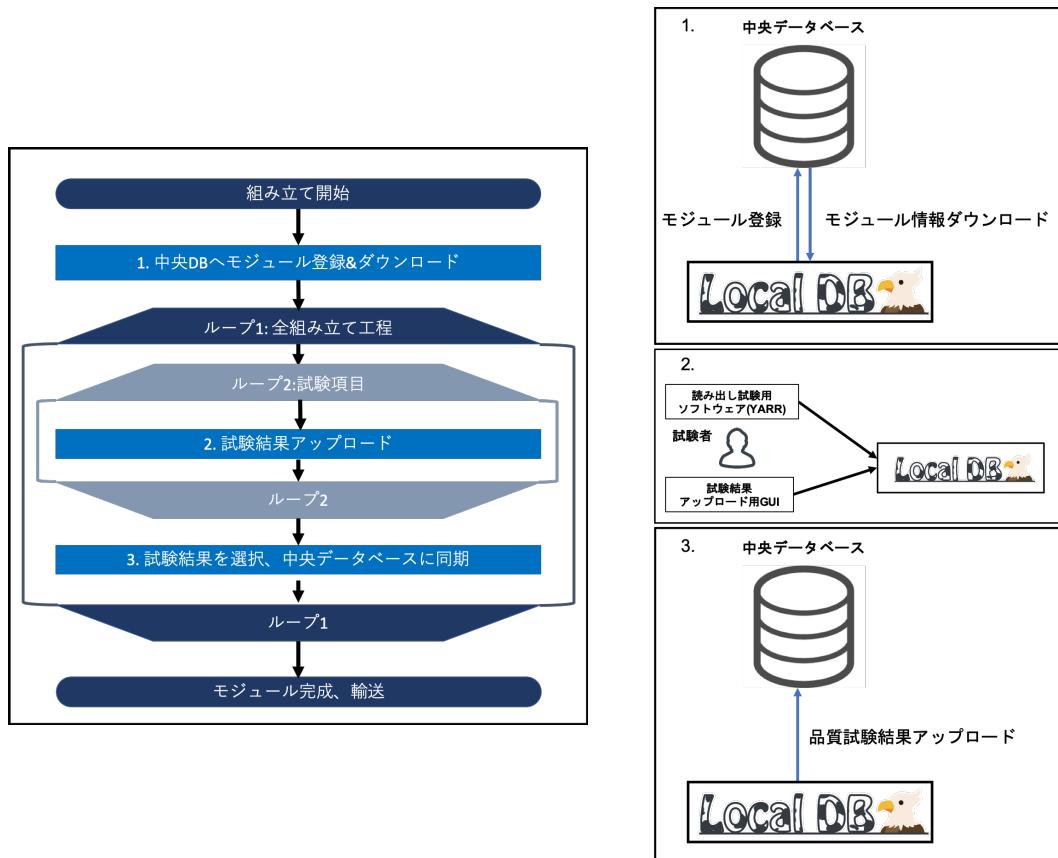


図 4.17 各モジュールにおけるデータベースシステム操作の流れ。モジュール組み立てにおけるデータベース操作の初めに、中央データベースにモジュール登録及びローカルデータベースへモジュール情報のダウンロードを行う(処理 1)。その後、ダウンロードしたモジュールに対して組み立て工程に応じた試験結果を生成、ローカルデータベースに保存する(処理 2)。各組み立て工程の終わりに試験結果の選択を行い、中央データベースに試験結果を同期する(処理 3)。

673 4.3.7 量産時におけるデータベース操作の流れの確立

674 量産時におけるデータベース操作の流れを確立した。以下に従い、モジュール組み立て時におけるデータ
675 管理がなされる。

- 676 1. 中央データベースへモジュール登録及び登録情報のダウンロード
- 677 2. 1で登録したモジュールに対して品質試験結果のローカルデータベースへのアップロード
- 678 3. ステージ毎に品質試験結果の登録と中央データベースへアップロード

679 流れのイメージを図 4.17 に示す。品質試験結果のアップロードは各組み立て工程毎に行う。ローカル
680 データベースで品質試験結果を組み立て工程毎にまとめて扱い、各モジュールの現組み立て工程を正確に
681 管理する目的がある。

682 全組み立て工程が終了すると、モジュールの情報及び品質試験結果が全て中央データベースへ同期され
683 ている状態となる。

684 この品質試験におけるデータベース操作の流れの確立に向けて、2月に CERN に行なったシステム
685 チュートリアルで操作の流れを試験した。ユーザに対する操作の流れの周知と機能確認を行うことができ
686 た。チュートリアルの概要は付録 C に記す。またシステムのドキュメント [22] を作成し、その中に具体

687 的なデータベース操作の流れを記述している。量産時に各機関が適切な流れでデータ管理ができるように
688 した。

689 データベース操作の流れの試験として、実際に品質試験のデモンストレーションを行い、各機能が正常
690 に動作するのかの確認を行なった。詳細を5章で述べる。

691 第5章

692 品質試験のデモンストレーション

693 品質試験項目のデモンストレーションを行った。今回のデモンストレーションでは、品質試験項目として読み出し試験を行い、データベース機能や試験の流れを確認した。この章の前半では開発したツールと試験で使用するソフトウェア、ハードウェアについて説明し、後にデモンストレーションの内容、各ソフトウェアの機能確認について述べる。

697 5.1 デモンストレーションと機能確認

698 上述したピクセル解析ツールを含む読み出し試験用ソフトウェアの機能確認を目的として、生産時における流れのデモンストレーションを行なった。その詳細について以下に示す。

700 5.1.1 用いたソフトウェア

701 試験で用いたソフトウェアをいかに示す。また、これらソフトウェアの概要を図 5.1 に示す。

- 702 • YARR(commit:6b3ffe92)
- 703 • MongoDB(version: v4.2.6)
- 704 • ローカル DB ウェブアプリケーション (tag: ldbtoolv1.4)
- 705 • 中央データベースとの同期ツール (tag: ldbtoolv1.4)
- 706 • ピクセル解析ツール (tag: v1.0.2)
- 707 • 時系列データ用データベース (InfluxDB[23](version: 1.8.0))
 - 708 – 時系列情報に特化したデータベース。このシステムにおいては温度、電圧など DCS 情報を時間情報と共に保存、管理するために用いる。
- 710 • InfluxDB 解析ソフト (Grafana[24](version: 5.1.0))
 - 711 – InfluxDB に保存された情報の解析、閲覧に用いる。ウェブブラウザー上で DCS データを閲覧することができる。
- 713 • 電源操作用ソフト
 - 714 – 電源を遠隔で操作し、モジュールに電圧を供給する。また電圧、電流値を取得し、InfluxDB にアップロードする。CERN で開発されている PySerial[25](commit:0d14fcdb) を改良し作成した。
- 717 • 温度読み出し用ソフト
 - 718 – GPIO 通信により取得できる ADC 値を取得、温度に変換し InfluxDB へアップロードする処

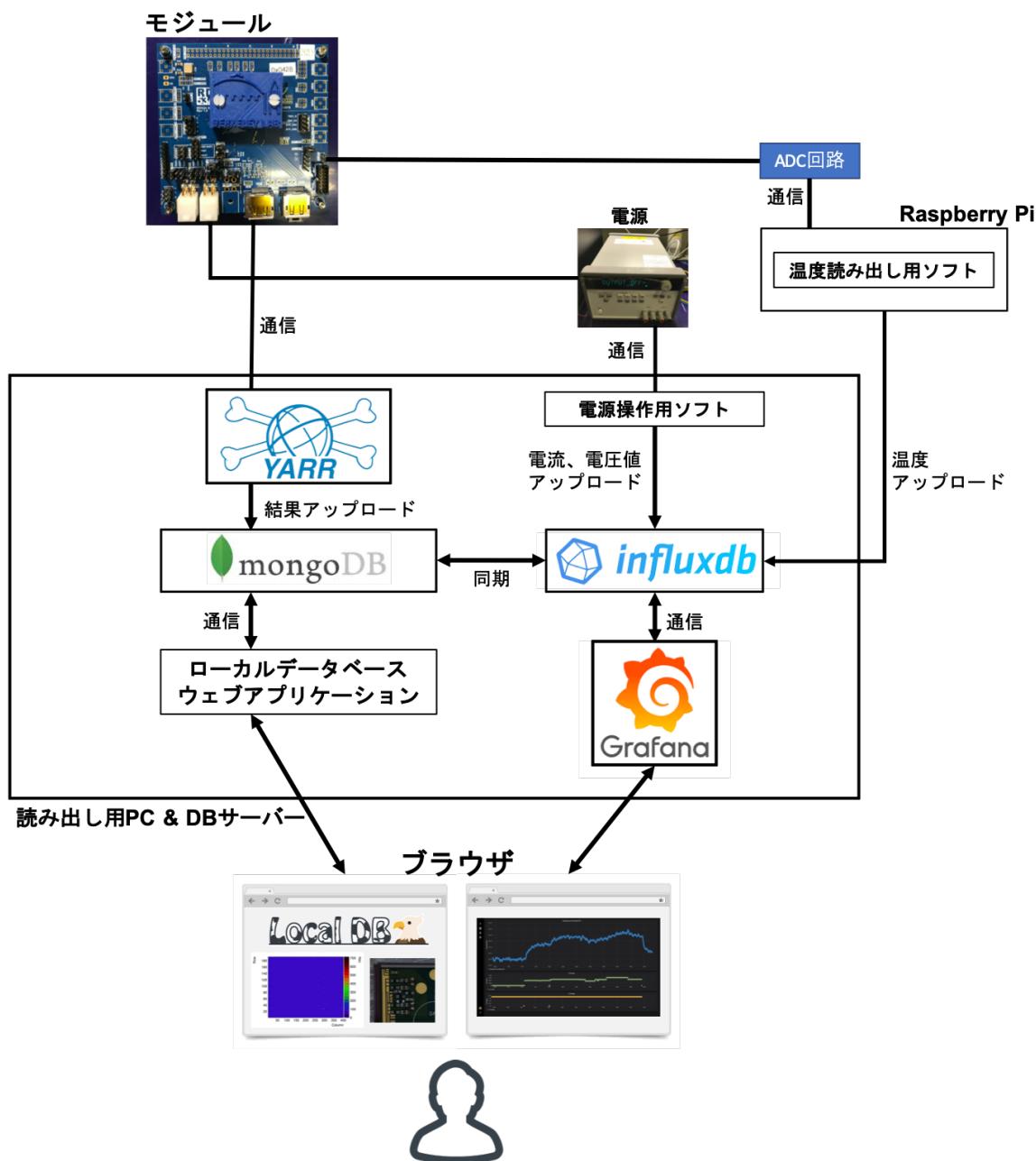


図5.1 読み出し試験に用いるソフトウェアの概要。FEチップの読み出しとそのデータ通信はYARRを用いて行われる。試験結果はMongoDBにアップロードされ、試験者はウェブアプリケーションを通じて結果を確認することができる。電源操作用ソフトを用いて電源のスイッチ、電圧、電流値の取得がなされ、取得値はInfluxDBに保存される。モジュール付属のサーミスタ読み出しシステムを用いてFEチップ付近の温度を読み出し、InfluxDBに保存する。InfluxDBに保存されたDCSデータはGrafanaを用いてブラウザ上で確認することができる。またMongoDBに同期されるため、ローカルデータベースアプリケーションを通して確認ができる。

理を作成したもの。RaspberryPi上に保存し、処理を実行。

5.1.2 用いたハードウェア

読み出し試験に用いたハードウェアについて、以下に詳細を記す。

RD53A シングルモジュール (RD53A Single Chip Card, SCC)[26]

今回読み出しに使うモジュールとして、研究室で所有している RD53A シングルモジュール (RD53A Single Chip Card, SCC) を使用した。SCC は試験用に作られた FE チップを一枚搭載するモジュールである。また今回使用したものはシリコンセンサーを持たない。SCC は FE チップ電源端子、データ転送端子をもち、読み出しを行う際はそれぞれ配線をする。FE チップ付近には NTC サーミスタを搭載していて、ボード上の端子からその抵抗値を取得することで温度を測定することができる。図 5.2 に写真を示す。



図 5.2 RD53A シングルモジュール (SCC)[26]。RD53A の FE チップを一枚搭載したモジュールである。図の中心部の青いカバーで囲われた中に FE チップが設置されている。このボード上にはデータ転送端子、電源端子、サーミスタ抵抗取得端子が設置されており、これらの端子に対応する配線をすることで読み出し試験のセットアップを組む。

モジュールサーミスタ温度読み出しシステム

モジュール付属サーミスタの抵抗値を取得し温度を測定するために、ADC、Raspberry Pi を用いた温度読み出しシステムを作成した。このシステムの中で扱った装置を表 5.1 に、回路図、実際に配線した様子を図 5.3 に示す。スクリプト上でサーミスタの抵抗値に対応する ADC 値を温度に変換することで読み出しを行っている。ADC 値は GPIO 通信 [27] により取得している。

電源

モジュールの FE チップに対する電圧供給に KEYSIGHT の E3646A 60W デュアル出力電源 [30](図 5.4) を用いた。

表 5.1 溫度読み出しシステムに使用した装置一覧。

装置	機種
10kΩ 抵抗	-
ADC	MCP3002[28]
RaspberryPi	Raspberry Pi 3 Model B Plus Rev 1.3[29]

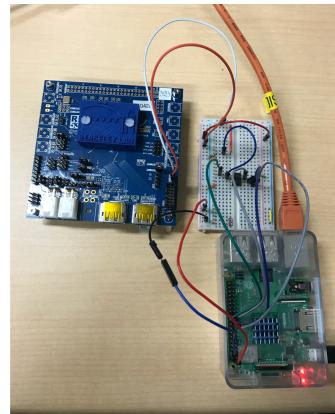
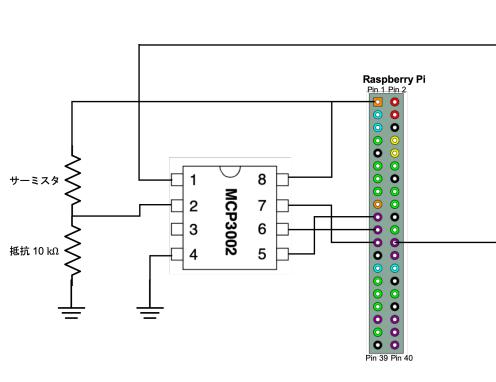


図 5.3 モジュール付属サーミスタを用いた温度読み出し回路。図は回路図（左図）と実際に配線し、読み出しを行っている様子（右図）を示している。抵抗、MCP3002、Raspberry Pi を用いて読み出し回路を作成した。抵抗と MCP3002 は右図のようにブレッドボード上に設置した。ADC と RaspberryPi は GPIO 通信 [27] を行うことで、ADC 値を取得している。



図 5.4 用いた電源。FE チップ電圧供給のための電源として KEYSIGHT の E3646A 60W デュアル出力電源 [30] を用いた。

737 FPGA ボード

738 FPGA ボードに XpressK7[31] を用いて、YARR ファームウェアを FEGA 上にプログラムし通信を行
739 なった。XpressK7 を図 5.5 に示す。

740 FMC-DisplayPort 変換カード

741 FPGA ボードには FMC 端子がついていて、これをモジュールを接続するためには FMC-DisplayPort
742 変換カードが必要となる。使用した変換カード（オハイオカード）を図 5.6 に示す。



図 5.5 使用した FPGA ボード (XpressK7[31])。中央に FMC 端子、右側に FPGA チップ、下側に PCI Express が配置されている。FPGA チップの上にはファンをついているため、この図では確認できない。



図 5.6 使用した FMC-DisplayPort 変換カード (オハイオカード)。図の右側に FMC 端子、左側に miniDisplayPort が 4 つ配置されており、FMC 端子を 4 つの miniDisplayPort に変換する。今回読み出す FE チップは 1 枚であるため、1 つの端子だけを用いる。

表 5.2 読み出しに使用した PC の性能。研究室で所有する PC を使用した。OS は centOS7 である。

CPU	Core	Thread	Clock speed[GHz]	Memory [GB]	Disk [GB]
Type					
Intel(R) Core(TM) i7-8700K CPU	6	12	3.7	16.18	214

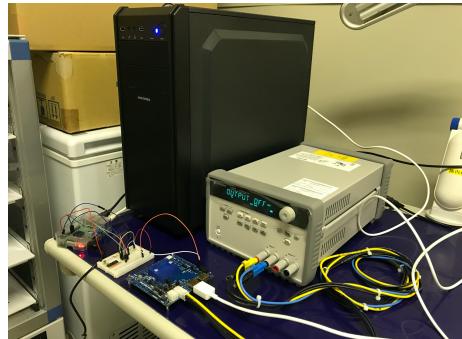
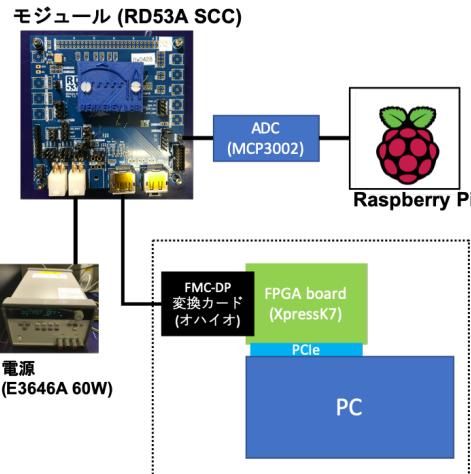


図 5.7 ハードウェアセットアップ。図はセットアップの概要 (左図) と実際に設置した様子 (右図) を示している。FE チップ読み出し装置、電源、サーミスタ読み出し装置をそれぞれ設置、配線し、読み出し操作、データ取得を行った。

743 PC

744 YARR ソフトウェアをインストールし、読み出しを行なった PC の性能を表 5.2 に示す。

745 セットアップ

746 読み出し試験に用いるハードウェアのセットアップを概要を図 5.7 に示す。

Latest Result								
Module Name	Chip Name	Current Stage	Test Type	User	Site	Date	Link	Tag
20UPGR00000001	20UPGFC9999999	MODULEWIREBONDING	PIXEL_FAILURE_TEST	hokuyama		2020/12/05 07:05:51	result page	Tag List

図 5.8 ダウンロードしたモジュール ID 確認画面。図の表において、今回登録した 20UPGR00000001 の ID を持つモジュールがローカルデータベースのウェブ上で確認できていることが分かる。また対応する FE チップの ID も確認することができる。

5.1.3 デモンストレーションの流れ

748 デモンストレーションで用いるため、中央データベースにおける登録 ID の定義方法 [32] に従い以下の
749 モジュール、FE チップを登録した。

750 モジュール ID 20UPGR00000001

751 FE チップ ID 20UPGFC9999999

752 今回のデモンストレーションではこれらの ID を用いて試験結果の紐付け等のデータベース操作を行う。
753 確認した機能を行った流れの順に以下に示す。

- 754 1. 中央データベースからモジュール情報のダウンロード.
- 755 2. 読み出し試験実施、結果をローカルデータベースに保存.
- 756 3. DCS 情報の取得、監視.
- 757 4. 試験結果検索.
- 758 5. 試験結果閲覧.
- 759 6. 結果選択とピクセル解析機能.
- 760 7. 中央データベースへ試験結果のアップロード

5.1.4 機能確認

762 読み出し試験を通して、各ソフトウェア機能が正しく動くことを確認した。詳細を以下に記す。

763 中央データベースからモジュール情報のダウンロード

764 ダウンロードし、ウェブアプリケーションで確認した。確認した画面を図 5.8 に示す。今回行った試験
765 結果はこのモジュールに紐付ける形でローカルデータベースに保存される。



図 5.9 DCS 情報のモニタリングの様子。図において全て横軸は時間であり、縦軸は上から温度(青)、電流(緑)、電圧(黄)を示す。それぞれのデータの監視が行えていることが分かる。温度に関して、電源オン、オフの付近で変化が大きいことが分かる。電流に関して、読み出し(チューニング)を行っている途中にも微小変化していることが分かる。

766 読み出し試験実施

767 以下の流れに沿って読み出しを行ない、結果をローカルデータベースに保存した。

- 768 1. デジタル回路読み出し (`std_digitalscan`)
- 769 2. アナログ回路読み出し (`std_analogscan`)
- 770 3. 調整前 Threshold 測定 (`std_thresholdscan`)
- 771 4. Threshold 調整
- 772 5. ToT 調整
- 773 6. Threshold 再調整
- 774 7. 調整後 Threshold 測定
- 775 8. ToT 測定 (`std_totscan`)
- 776 9. ノイズ測定 (`std_noisescan`)

777 また読み出し試験を通して DCS 情報は InfluxDB を用いて監視し、試験結果と同様にローカルデータ
778 ベースに保存した。

779 DCS 情報の監視

780 読み出し試験は、DCS 情報を監視しながら行った。それぞれの値は対応するソフトウェアを用いて
781 InfluxDB にアップロードした。その値を Grafana を使って監視をした。その様子を図 5.9 に示す。

The screenshot shows a table titled "Test Data" with columns: Module Name, Chip Name, Test Type, User, Site, Date, Link, and Tag. The table contains 16 rows of data, each representing a different test run. The "Link" column for all rows contains the text "result page".

Test Data							
Module Name	Chip Name	Test Type	User	Site	Date	Link	Tag
20UPGR00000001	20UPGFC999999	std_digitalscan	hokuyama	lazulite	2020/11/13 19:49:40	result page	
20UPGR00000001	20UPGFC999999	std_digitalscan	hokuyama	lazulite	2020/11/13 19:49:00	result page	
20UPGR00000001	20UPGFC999999	std_noisescan	hokuyama	lazulite	2020/11/13 19:41:07	result page	
20UPGR00000001	20UPGFC999999	std_totscan	hokuyama	lazulite	2020/11/13 19:40:57	result page	
20UPGR00000001	20UPGFC999999	std_thresholdscan	hokuyama	lazulite	2020/11/13 19:38:59	result page	
20UPGR00000001	20UPGFC999999	syn_tune_globalthreshold	hokuyama	lazulite	2020/11/13 19:38:11	result page	
20UPGR00000001	20UPGFC999999	syn_tune_globalpreamp	hokuyama	lazulite	2020/11/13 19:37:44	result page	
20UPGR00000001	20UPGFC999999	syn_tune_globalthreshold	hokuyama	lazulite	2020/11/13 19:36:54	result page	
20UPGR00000001	20UPGFC999999	lin_retune_pixelthreshold	hokuyama	lazulite	2020/11/13 19:36:35	result page	
20UPGR00000001	20UPGFC999999	lin_tune_globalpreamp	hokuyama	lazulite	2020/11/13 19:36:13	result page	
20UPGR00000001	20UPGFC999999	lin_retune_pixelthreshold	hokuyama	lazulite	2020/11/13 19:35:55	result page	
20UPGR00000001	20UPGFC999999	lin_retune_globalthreshold	hokuyama	lazulite	2020/11/13 19:35:35	result page	
20UPGR00000001	20UPGFC999999	lin_tune_pixelthreshold	hokuyama	lazulite	2020/11/13 19:35:09	result page	
20UPGR00000001	20UPGFC999999	lin_tune_globalthreshold	hokuyama	lazulite	2020/11/13 19:34:53	result page	
20UPGR00000001	20UPGFC999999	diff_tune_pixelthreshold	hokuyama	lazulite	2020/11/13 19:34:36	result page	

↓

"hokuyama std_totscan"
で検索

The second screenshot shows the same interface after a search for "hokuyama std_totscan". The search results table now only contains one row of data, corresponding to the entry in the first screenshot.

Test Data							
Module Name	Chip Name	Test Type	User	Site	Date	Link	Tag
20UPGR00000001	20UPGFC999999	std_totscan	hokuyama	lazulite	2020/11/13 19:40:57	result page	

図 5.10 検索機能確認の様子。図は検索実行前の試験結果一覧（上図）と実行後（下図）を示す。図の例では”hokuyama std_totscan”の 2 つのキーワードで検索を行っており、実行後は対応する試験結果 1 つが表示されていることが分かる。この試験結果について試験実施者は”hokuyama”、試験項目は”std_totscan”であるため、検索機能が正常に動いていることが分かる。

782 検索機能

783 検索機能の確認を行い、正常に使用できることを確認した。検索機能実行の様子を図 5.10 に示す。

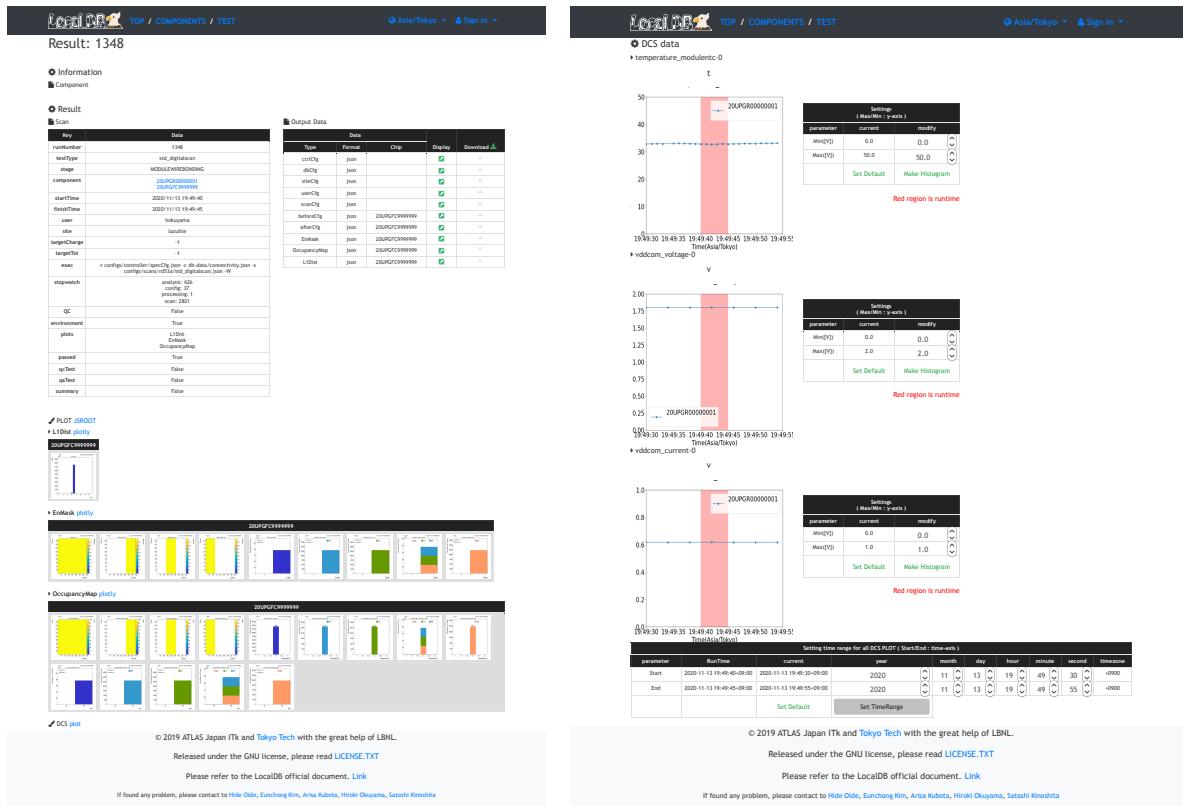


図 5.11 試験結果の閲覧。図は試験結果（左図）と試験におけるDCSデータのグラフ（右図）を示しており、上から温度、電圧、電流となっている。図はstd_digitalscanの結果であり、試験情報及び結果のグラフが確認できる。また右図よりDCSデータも正常にローカルデータベース上に保存され、表示されていることが分かる。

784 試験結果閲覧

785 ウェブアプリケーションを用いて、試験結果を閲覧した。その様子を図 5.11 に示す。

786 結果選択とピクセル解析

787 読み出し結果を選択し、ピクセル解析を行なった。結果選択画面を図 5.12 に示す。このデモンスト
788 レーションにおける不良評価基準は 3 章に述べた表 3.1 の中から、現時点でシステムに実装している以下
789 の項目を抜粋した。

- 790 1. Digital Dead
791 2. Digital Bad
792 3. Analog Dead
793 4. Analog Bad
794 5. Tuning Failed
795 6. Tuning Bad for Threshold
796 7. Tuning Bad for ToT
797 8. High ENC
798 9. Noisy

799 解析結果を図 5.13、それぞれの評価基準における不良ピクセルの分布を図 5.14 に示す。

800 試験結果アップロード

801 読み出し試験の結果を中央データベースにアップロードし、情報が正しくアップロードされていること
802 を確認した。図 5.15 に中央データベースのウェブページを示す。結果ファイルや解析結果が正しくアッ
803 プロードされていることが分かる。

LocalDB TOP / COMPONENTS / TEST

Selection Page
→Select scan results for QC result in this stage.

Component: 20UPGR00000001
Current Stage: MODULEWIREBONDING

Please select one result for each test items.

►std_digitalscan

Test Data										
run Number	Module Name	Chip Name	Stage	Test Type	User	Site	Date	Link	Tag	Select
1392	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	tokyo_institut e_of_technology	2020/12/08 22:35:47	result page		<input type="radio"/>
1391	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	tokyo_institut e_of_technology	2020/12/08 22:29:39	result page		<input type="radio"/>
1351	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	tokyo_institut e_of_technology	2020/12/03 14:54:43	result page		<input type="radio"/>
1350	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	tokyo_institut e_of_technology	2020/11/30 19:57:19	result page		<input type="radio"/>
1348	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	lazulite	2020/11/13 19:49:40	result page		<input type="radio"/>
1347	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	lazulite	2020/11/13 19:49:00	result page		<input type="radio"/>
1328	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	lazulite	2020/11/13 19:31:02	result page		<input checked="" type="radio"/>
1327	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	lazulite	2020/11/13 19:29:24	result page		<input type="radio"/>
1326	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_digitalscan	hokuyama	lazulite	2020/11/13 19:28:38	result page		<input type="radio"/>

►std_analogscan

Test Data										
run Number	Module Name	Chip Name	Stage	Test Type	User	Site	Date	Link	Tag	Select
1329	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_analogscan	hokuyama	lazulite	2020/11/13 19:31:13	result page		<input checked="" type="radio"/>

►std_thresholdscan

Test Data										
run Number	Module Name	Chip Name	Stage	Test Type	User	Site	Date	Link	Tag	Select
1344	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_threshold scan	hokuyama	lazulite	2020/11/13 19:38:59	result page		<input checked="" type="radio"/>
1330	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_threshold scan	hokuyama	lazulite	2020/11/13 19:31:24	result page		<input type="radio"/>

►std_totscan

Test Data										
run Number	Module Name	Chip Name	Stage	Test Type	User	Site	Date	Link	Tag	Select
1345	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_totscan	hokuyama	lazulite	2020/11/13 19:40:57	result page		<input checked="" type="radio"/>

►std_noisescan

Test Data										
run Number	Module Name	Chip Name	Stage	Test Type	User	Site	Date	Link	Tag	Select
1346	20UPGR00000001	20UPGFC999999	MODULEWIRE BONDING	std_noisescan	hokuyama	lazulite	2020/11/13 19:41:07	result page		<input checked="" type="radio"/>

Proceed

[Back to the component page](#)

© 2019 ATLAS Japan ITk and [Tokyo Tech](#) with the great help of LBNL.
Released under the GNU license, please read [LICENSE.TXT](#)
Please refer to the LocalDB official document. [Link](#)
If found any problem, please contact to [Hide Oide](#), [Eunchong Kim](#), [Arisa Kubota](#), [Hiroki Okuyama](#), [Satoshi Kinoshita](#)

図 5.12 読み出し試験結果の選択。図は読み出し試験結果選択画面を表す。読み出し試験実施後、ピクセル解析と中央データベースとの同期を実行するために試験結果を選択する必要がある。図では std_digitalscan、std_analogscan、std_thresholdscan、std_totscan、std_noisescan の 5 項目を選択している。

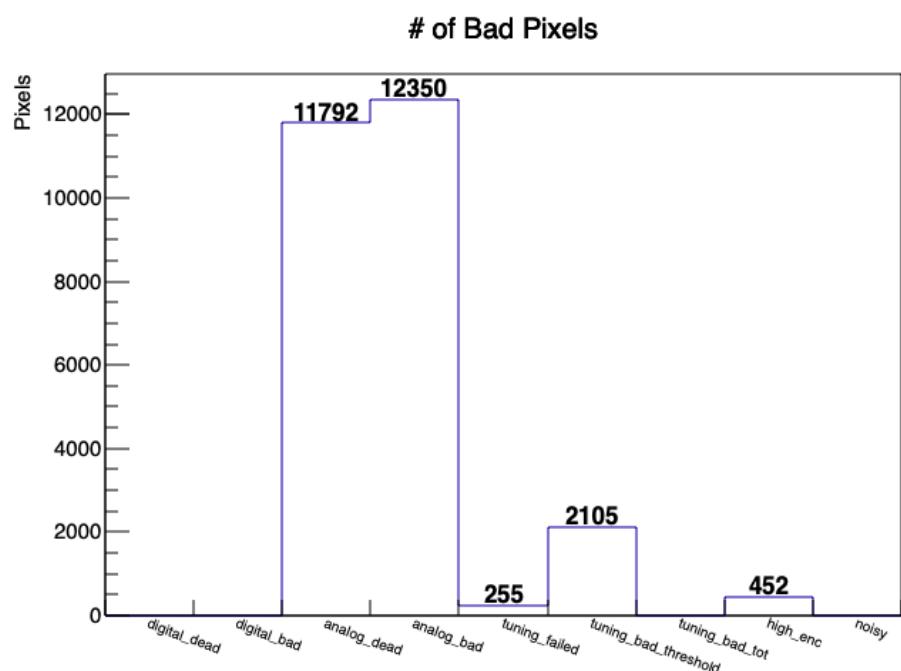


図 5.13 ピクセル解析結果。図において横軸は評価基準、縦軸は該当するピクセル数を表す。
analog_dead, analog_bad の割合が多いいため、アナログ回路読み出しに失敗しているピクセルが多いことが読み取れる。

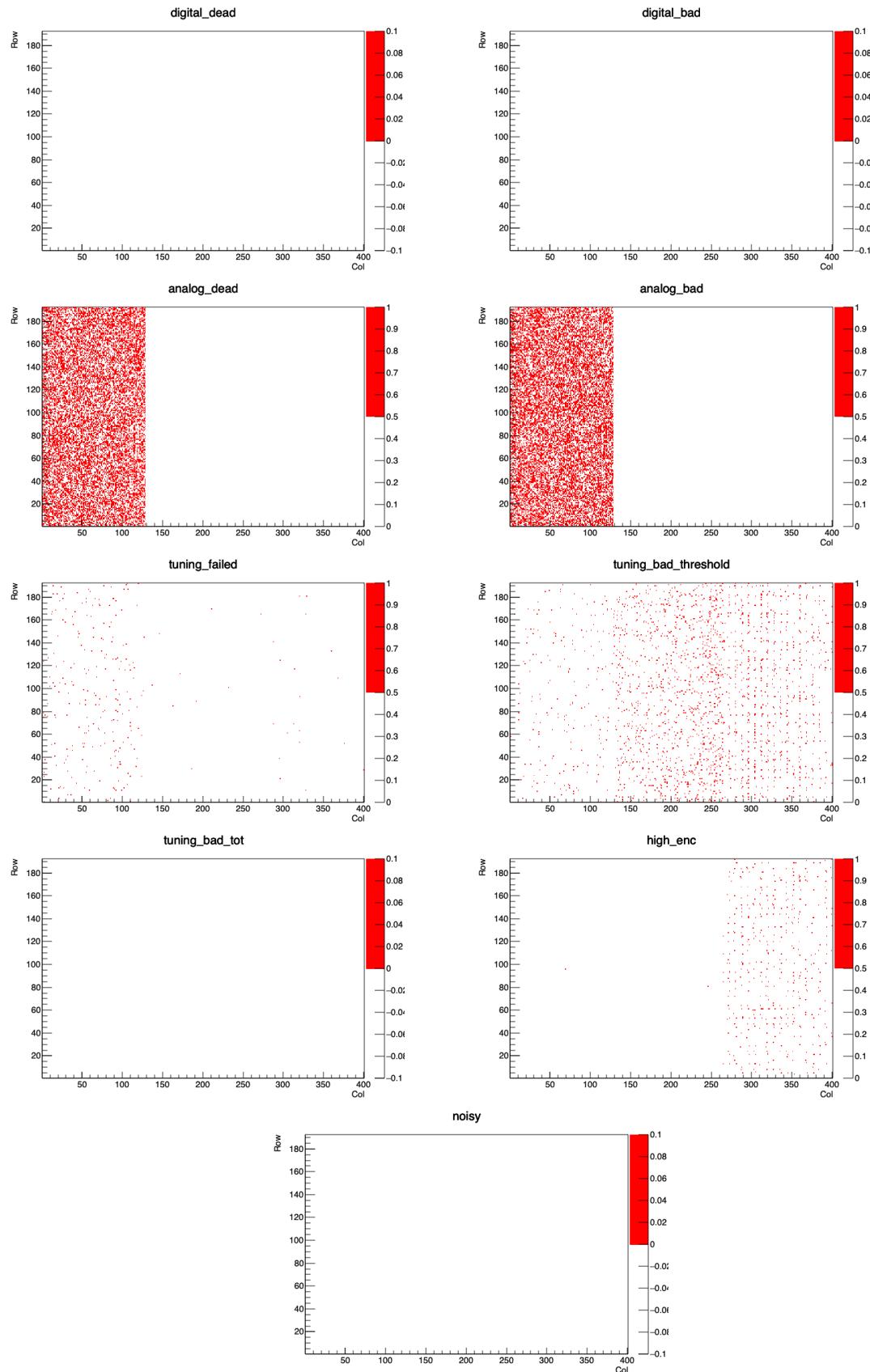


図 5.14 各評価基準における不良ピクセルの分布。各図は二次元ヒストグラムであり、横軸が FE チップにおける各ピクセルの列番号、縦軸が行番号を示しており、赤い領域が不良ピクセルを表している。図 5.13 においてアナログ回路読み出しに失敗しているピクセルが多いことがわかったが、その不良ピクセルは synchronous フロントエンド上に存在していることが分かる。また Threshold 調整に失敗しているピクセル (tuning_bad_threshold) は全体的に分布していることが分かる。

The screenshot shows the 'Test Run Details' page for a 'ModuleQC::PixelFailureTest'. The top navigation bar includes 'ATLAS ITk Production Database' and 'Test' tabs, and a user profile for Hiroki Okuyama.

Basic Properties:

- INSTITUTE: Tokyo Institute of Technology TITECH
- RUN NUMBER: ... ✓
- TEST DATE: 13.11.2020 19:31
- TEST TYPE: ModuleQC::PixelFailureTest PIXEL_FAILURE_TEST
- UPLOADED: Hiroki Okuyama

Associated Components:

- Front-end Chip - RD53A 20UPGFC999999
- Tested at stage: Test after bump bonding on module

Properties:

- QC stage: MODULEWIREBONDING

Results:

Category	Count
Noisy	0
High ENC	452
Tuning Bad ToT	0
Tuning Bad Threshold	2105
Tuning Failed	255
Analog Bad	12350
Analog Dead	11792
Digital Bad	0
Digital Dead	0

Defects:

Attachments:

- std_digitalscan_datafiles.zip
- std_digitalscan_configfiles.zip
- std_analogscan_datafiles.zip
- std_analogscan_configfiles.zip
- std_thresholdscan_datafiles.zip
- std_thresholdscan_configfiles.zip
- std_totscan_datafiles.zip
- std_totscan_configfiles.zip
- std_noisescan_datafiles.zip
- std_noisescan_configfiles.zip

Comments:

© Copyright Unicorn College s.r.o. 2020
cernitkpdg01-0.18.3

図 5.15 中央データベースにおける読み出し試験及びピクセル解析結果画面。読み出し試験及びピクセル解析結果を中央データベースにアップロードした。図は中央データベースのウェブページを示しており、アップロードされた試験結果画面である。図の中央部に各評価基準における不良ピクセルの数、下部に各試験項目ごとの結果ファイルをまとめた Zip ファイルが表示されていることが分かる。試験結果のアップロードが正常に完了し、中央データベース上で結果を確認できることが分かる。

804 第6章

805 ローカルデータベースにおける検索機能 806 とその処理時間

807 モジュール組み立てにおいて、読み出し試験は複数回行われ、一回の試験で行う項目も複数存在する。
808 そのため、生産時には試験結果が数多くデータベースにアップロードされることになる。4章で述べたよ
809 うに、任意のタイミングで必要な結果を取得できる検索機能を実装した。詳細について以下に示す。

810 6.1 実装方法

811 今回の実装では、一般的にウェブで用いられているフリーワードの検索エンジンのような機能を実装し
812 ようと考えた。ユーザの操作を最小限にし、直感的かつ柔軟な検索ができるようにするためにある。
813 読み出し試験において、対象とする検索キーワードを以下の項目に絞った。システムにおいて、アップ
814 ロードされた試験結果に関わるデータベース内の情報は後から編集する機能はサポートしない方針を取っ
815 ている。品質試験の結果が後から上書きされると、結果の信頼性を失うと考えているからである。そのた
816 め、ユーザが対象としたい検索キーワードは以下の項目に限られ、検索キーワードとしてサポートすれば
817 十分であると考えた。

- 818 • モジュール及びFEチップのID.
- 819 • 読み出し試験項目(例:std_digitalscan)
- 820 • 読み出し試験者.
- 821 • 読み出し試験場所.
- 822 • 試験日時.
- 823 • タグ機能を用いてつけられたタグ.

824 そこで実装方法として、以下の2つを考えた。

- 825 1. 各試験に関する情報をプログラム上で配列に保持し、検索キーワードが含まれるかを確認する方法.
 - 826 2. 各試験に関する情報を持つドキュメント、コレクションを予め作成、それを参照し検索を行う方法.
- 827 これらについて以下で詳細を説明する。

828 6.1.1 方法 1: Python リストを用いた一致確認

829 Python リストを使う実装の場合、以下のような流れで検索処理を行う。

- 830 1. ユーザが検索キーワードを入力し、処理を実行.
831 2. アップロードされた全ての読み出し試験結果に関する情報を取得.
832 3. 各試験結果に関する情報を Python リストに保持、検索キーワードとの一致を確認、試験を選別.
833 4. ブラウザに送信.

834 アルゴリズムのイメージを図 6.1 に示す。この方法では、データベース内の試験結果とアプリケーションの関数内だけで全ての処理を行うことが可能なため、シンプルな実装方法であると言え、直感的に始めに思いつく方法であった。

837 しかしこの方法を試験実装したところ、ドキュメント数の増加に対して検索処理時間を大きく要して
838 しまう問題が発生した。図 6.2 のようにデータベース内の構造は複数のコレクションを跨いで情報を保
839 持しているため、試験結果全てに対してリアルタイムでこの処理を行うと、検索時間が大きくかかる
840 しまう。このシステムのデータ構造においては、表 4.1 において各試験結果の情報を保存する testRun、
841 component と testRun の関係を保存する componentTestRun の構造による遅延であると考えられる。
842 試験結果数を n とすると、testRun が n 、componentTestRun がそれぞれ $O(n)$ のドキュメント数を持つ
843 ことになる。これらのドキュメントを全て検索し、一致確認を行うと全体で $O(n^2)$ の時間がかかる。イ
844 メージを図 6.3 に示す。この実装方法について、ドキュメント数と処理時間の関係を測定したところ、図
845 6.4 のようになり、確かにドキュメント数に対して 2 次的に増加していることが分かる。測定の詳細につ
846 いては後述する。

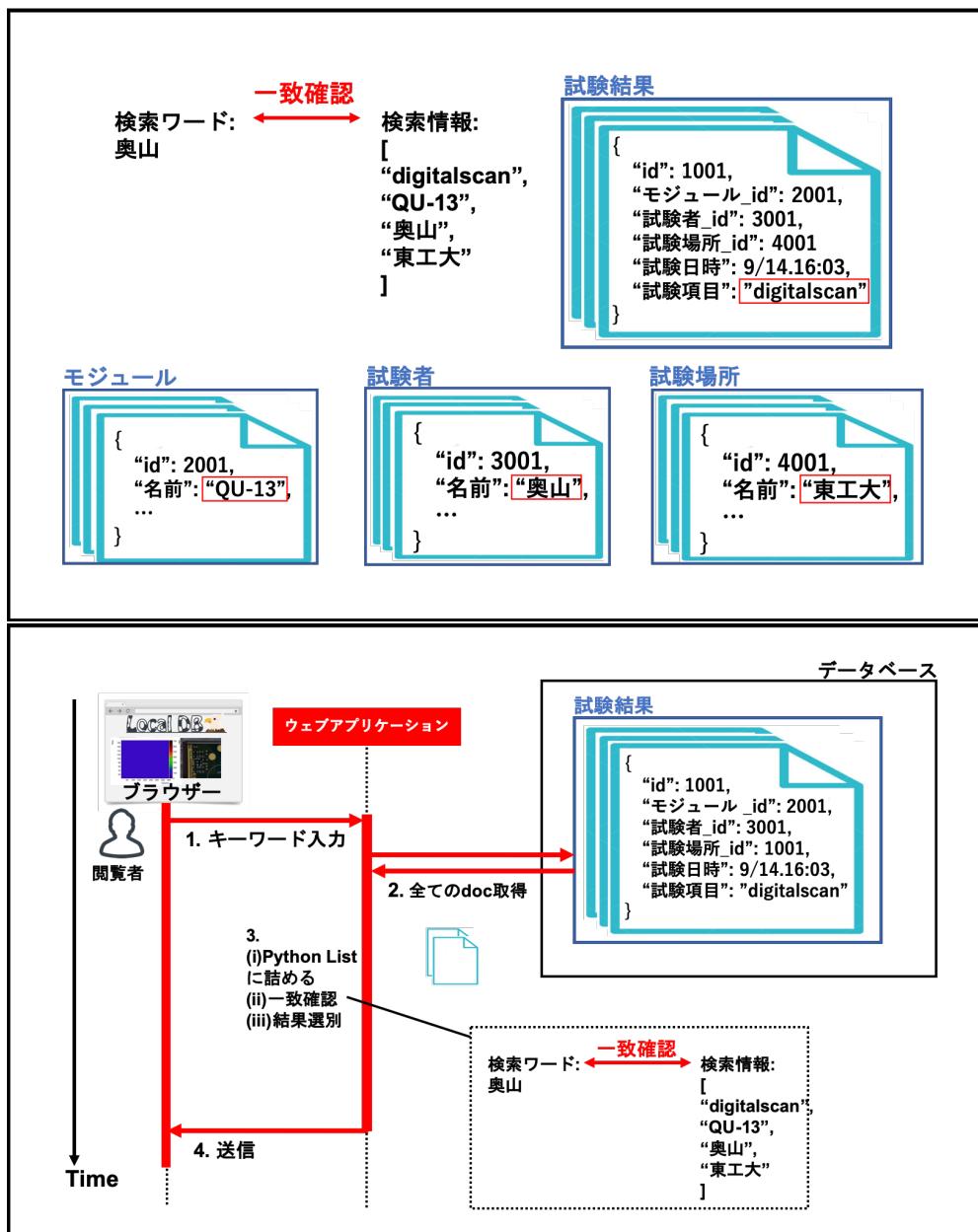


図 6.1 検索機能実装方法 1:Python リストを用いた場合の検索処理のイメージ図。上図は各コレクションと保存されている情報の例を示しており、下図は実際に検索を行った時の処理の流れを表している。上図中の赤枠で囲われた情報のように検索対象となる名前の情報は複数のコレクションにまたがって保存されている。各試験結果に対してこれらの情報を集めリスト内に保持し、各要素とキーワードとの一致を確認することで検索処理を行う。

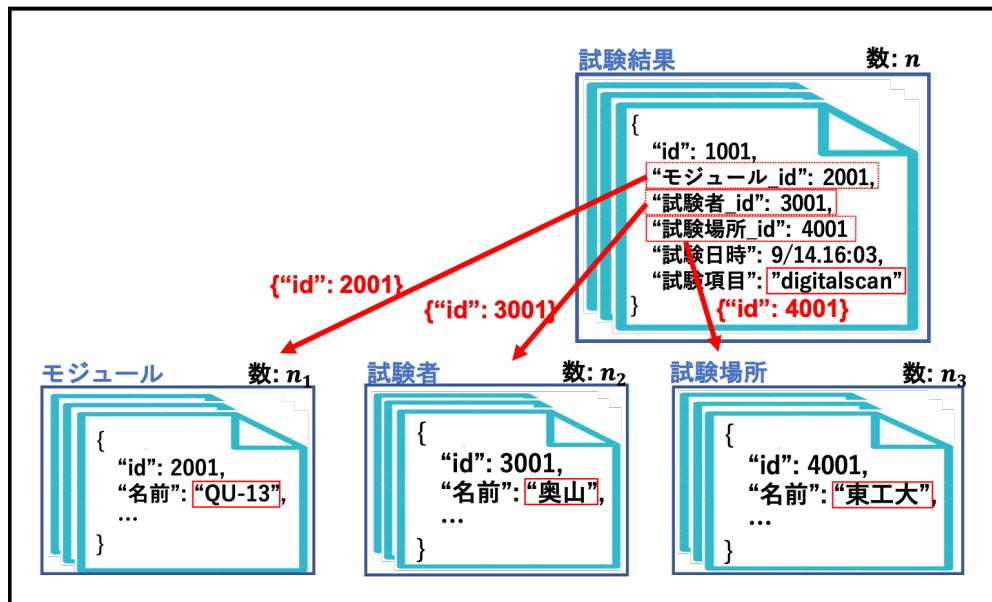


図 6.2 検索機能実装方法 1 の問題点のイメージ図。図 6.1 でも述べたように、検索対象となる情報はこの図のように複数のコレクションにまたがって保存される。そのため、コレクションを超えて検索を行うような場合は試験結果の数以上に、処理に時間がかかるてしまう。この図の例の場合、あるコレクション検索にかかる時間がドキュメント数に対して線形とすると、 $O(n * (n_1 + n_2 + n_3))$ の処理時間がかかると考えられる。

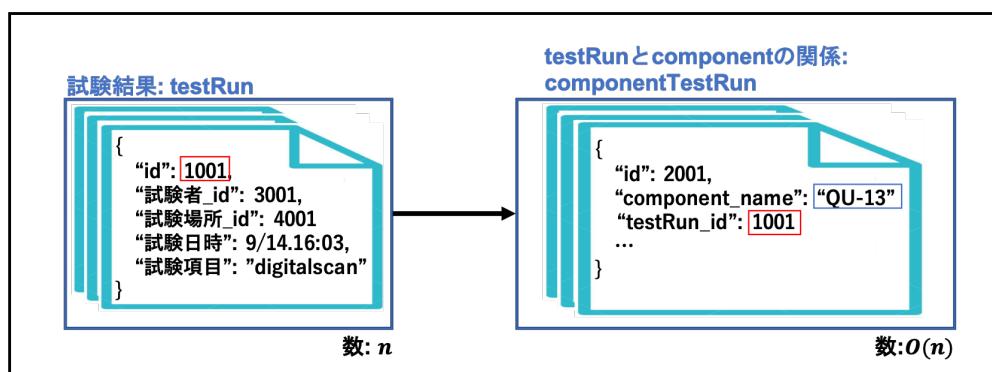


図 6.3 検索処理時間のボトルネックとなっているデータ構造の図。読み出し試験におけるデータ構造(図 4.3)の 1 つに、各試験結果情報を保存する `testRun`、`component` と `testRun` の関係性を保持する `componentTestRun` という構造が存在する。試験結果数を n とすると、`testRun` のドキュメント数は n 、`componentTestRun` の数は試験数と `component` の数の積となるため $O(n)$ となる。結果的に検索処理に $O(n^2)$ の時間がかかるてしまう。

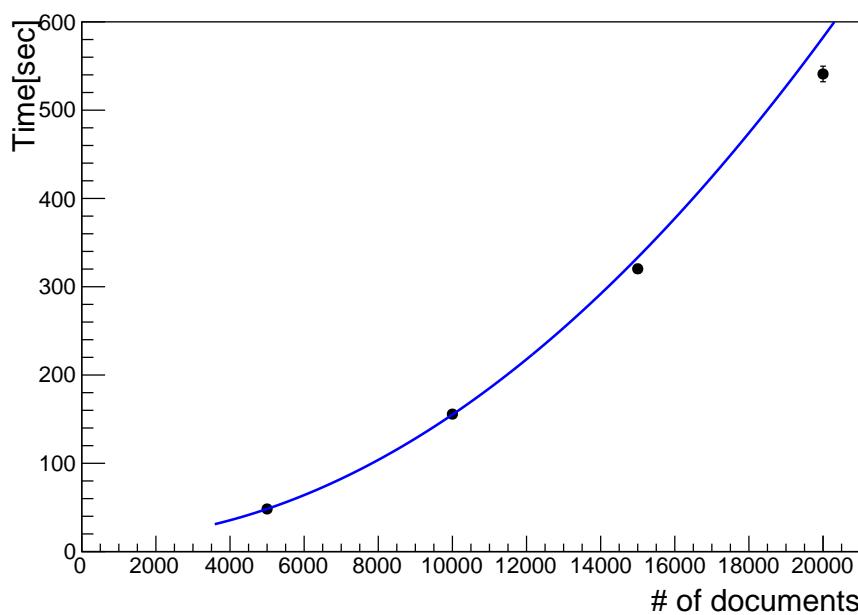


図 6.4 方法 1 における検索処理速度測定結果。横軸が試験結果のドキュメント数、縦軸が処理時間を表している。図 6.3 で述べたように、方法 1 の検索処理では試験結果数 n に対して $O(n^2)$ の検索時間がかかるため、試験結果のドキュメント数に対して処理時間は二次関数になっていることがわかる。近似関数や生産時における処理時間の見積もりに関しては後述する。

847 6.1.2 方法2: 検索情報を持つドキュメントを作成、使用

848 検索機能を改善するため方法2を考案し、実装を行った。改善の方法として、読み出し試験のアップ
849 ロードシステム及びウェブアプリケーションでの結果確認システムは既に使われていたため、データ構造
850 及び使用しているPythonフレームワークの変更はせずに処理時間を改善することを試みた。

851 改善策として、検索キーワードを別のドキュメントに予め保持しておき、処理実行時にはそれを参照す
852 ることで検索を行う方法を考案し、試験を行った。アルゴリズムのイメージを図6.5に示す。検索情報コレクションに入
853 ションに入るドキュメント数は、試験結果数と同じである。なお、このコレクションはウェブアプリケーションの立ち上げ時に生成するシステムとした。そのため試験結果数に対する処理時間は $O(n)$ と考えられ、方法1に比べて検索コストを削減できると考えた。

856 以下に検索情報のドキュメントの例をリスト6.1に示す。

ソースコード6.1 検索情報コレクションに入るドキュメントの例。このように試験結果のID、試験日時、検索対象となる名前情報が保存される。

```
857 1 {  
858 2     "_id" : ObjectId("5fd489f60e2ca70557e44a8b"),  
859 3     "runId" : "5fc4d027b1ef7c6297c91040",  
860 4     "timeStamp" : ISODate("2020-11-30T10:57:19Z"),  
861 5     "data" : [  
862 6         "20UPGR00000001",  
863 7         "20UPGFC99999999",  
864 8         "std_digitalscan",  
865 9         "hokuyama",  
866 10        "tokyo_institute_of_technology",  
867 11        "2020/12/09"  
868 12    ]  
869 13 }
```

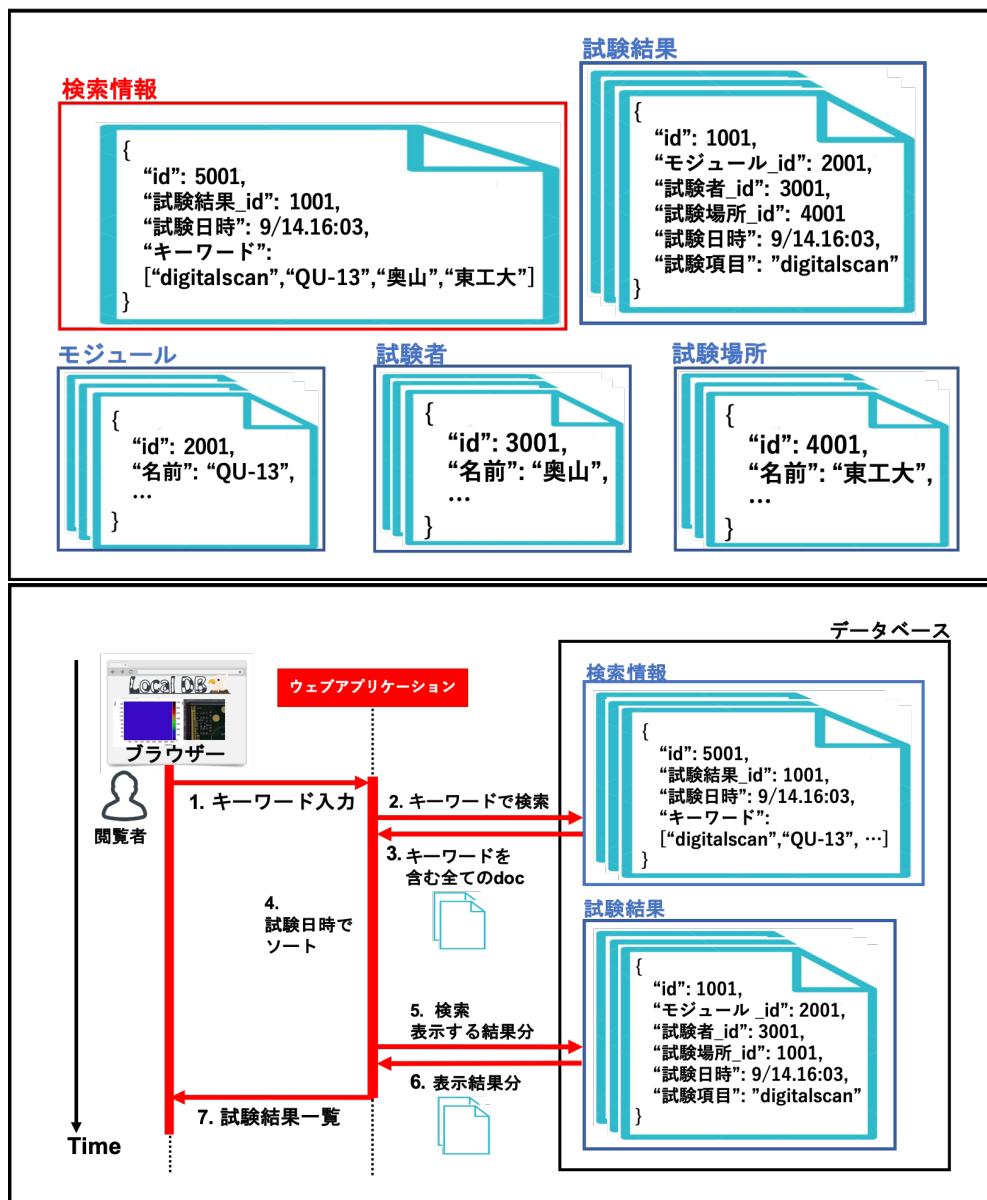


図 6.5 検索機能実装方法 2:検索キーワード専用コレクションを用いた場合のイメージ図。上図は各コレクションと保存されている情報の例を示しており、下図は実際に検索を行った時の処理の流れを表している。方法 2 では上図の赤枠で囲っている検索情報のコレクションを新たに設け、これを参照することで検索処理を行う。このコレクション内には各試験結果に対応したドキュメントが 1 つ保存され、全検索情報と試験結果の ID を持つものとなっている。処理の流れとしては下図のように、検索情報のコレクションよりキーワードに対応する試験結果を抽出する処理と、表示の際に必要な情報を試験結果のコレクションから取得する処理の 2 つに分かれた構造となっている。このような処理をすることにより、各検索処理にかかる時間は試験結果数に対して $O(n)$ になると考えられる。

表 6.1 測定に使用したノート PC(MacBookAir(13-inch,2017)) の性能。検索処理時間の測定に個人的に使用しているノート PC を使用した。

種類	CPU	Type	Core	Thread	Clock speed[GHz]	Memory [GB]	Disk [GB]
MacBookAir(13-inch,2017)	Intel Core i5	2	4	1.8		8	256

表 6.2 検索機能処理時間測定の詳細。測定を行った試験結果数、回数、キーワード、検索モード、検索情報の詳細を示している。

試験結果数	5000, 10000, 15000, 20000
測定回数	各測定点に対して 20 回
検索キーワード	okuyama
検索モード	部分一致
各試験結果が持つ検索情報	全試験結果に対して同じ、以下に詳細
検索情報一覧	モジュール名: 20UPGR10000005 FE チップ名: 20UPGTU9000000 試験項目: std_digitalscan 試験者: okuyama 試験場所: default_host 試験日時: 2020/12/06

6.2 処理時間測定

考案した方法を実装し、検索処理時間の測定を行った。詳細を以下に示す。また方法 1 において行った処理時間測定も同様の条件で行った。

使用した装置

測定には個人的に使用しているノート PC(MacBookAir(13-inch,2017)) を用いた。性能を表 6.1 に示す。

測定内容

コマンドプロンプトから以下のコマンドを実行し、ある試験結果ページのリクエストに対するアプリケーションのレスポンス時間を測定した。ここでは検索キーワードは”okuyama”とし、検索モードは部分一致としている。実際にアプリケーションを使用する際には、ブラウザに一覧表示をする時間が今回の測定時間に加算されることになる。

```
1 curl "http://127.0.0.1:5000/localdb/scan?keywords=okuyama&match=partial" -o /dev/null -w "%{time\_total}\n" 2> /dev/null -s
```

その他測定に関する詳細を表 6.2 に示す。

各測定点に対して平均値、標準偏差を算出し、フィッティングを行った。

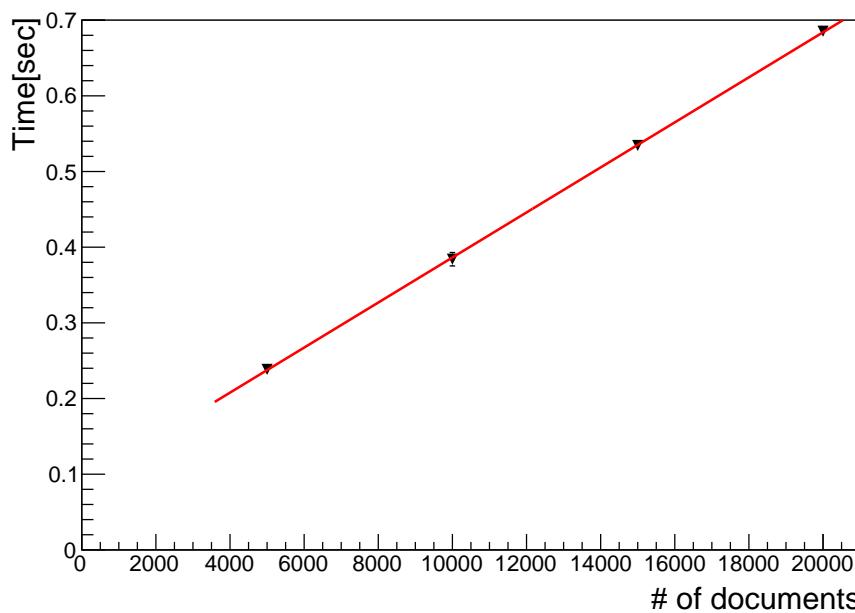


図 6.6 方法 2 における検索処理時間測定結果。横軸が試験結果のドキュメント数、縦軸が処理時間を表している。線形性を示していることが確認でき、検索実行時における処理時間は方法 1 に比べて優れている。

889 測定結果

890 方法 2 を用いて得られた結果を図 6.6 に示す。また方法 1 の結果に関しては、既に図 6.4 で示した。方
891 法 1、方法 2 で得られた近似関数を式 6.2、6.3 に示す。方法 1 に対して、方法 2 はアルゴリズムの改善が
892 見られる。

$$y = ax^2 + b \quad (6.1)$$

$$a = (1.4 \pm 0.0) \times 10^{-6}$$

$$b = 13 \pm 0$$

$$y = ax + b \quad (6.2)$$

$$a = (3.0 \pm 0.1) \times 10^{-5}$$

$$b = (8.9 \pm 0.8) \times 10^{-2}$$

893 現在は方法 2 で検索機能を実装し、サービスの 1 つとして提供している。

894 方法 2 では、アプリケーションの起動時に検索情報コレクションを生成するアルゴリズムとしている。
895 ここで検索情報コレクションの生成にかかる処理時間として、ドキュメント数に対するアプリケーション
896 の起動時間を測定した。結果を図 6.7 に示す。基本的には方法 1 と同じ操作を行うことで検索情報を収集
897 し、ドキュメントを作成する。そのため、処理時間は $O(n^2)$ となる。この処理を前もって行なうことで、
898 検索実行時の処理性能を上げている。

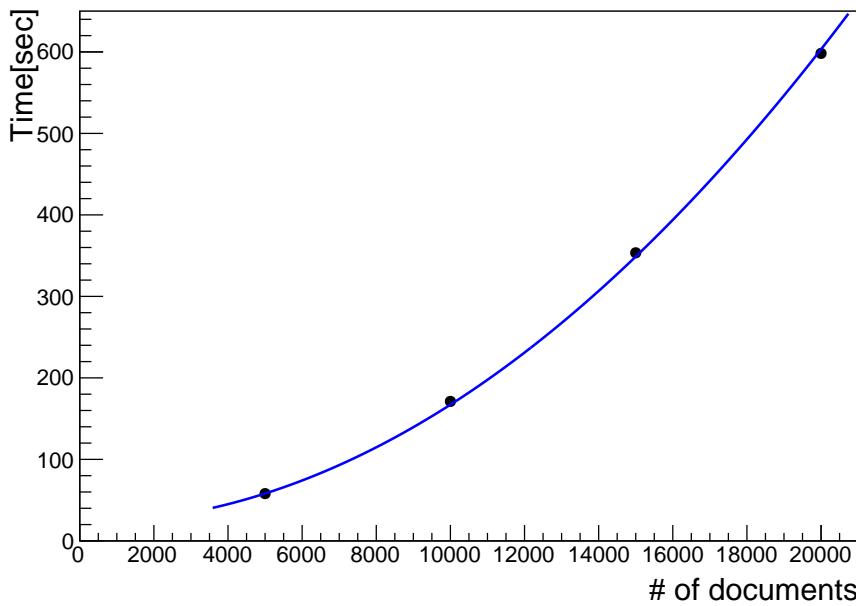


図 6.7 方法 2 における検索情報コレクションの生成時間測定結果。検索情報コレクションの生成にかかる処理時間として、ドキュメント数に対するアプリケーションの起動時間を測定した。横軸が試験結果のドキュメント数、縦軸が時間を示している。検索情報コレクションの生成には、方法 1 と同様の処理を行なっているため $O(n^2)$ となっているのが分かる。

899 生産時における検索時間の見積もり

900 各方法について、生産時における処理時間の見積もりを行う。簡単のため今回使用したデバイスと生産
901 時に使うサーバーの性能差は無視する。ここでデータベースで管理するモジュール数は日本が最多とし、
902 その数を予定している 2,000 とする。保存する読み出し試験数は 3 章で述べたように、1 つのモジュール
903あたり 42 とする。全ての生産が終了した際の検索処理時間を見積もる。上で得られた関係式を用いて検
904索処理実行時間は方法 1、2 に対して式 6.3、6.4 のように見積もることができる。

$$\{(1.4 \pm 0.0) \times 10^{-6}\} \times (2000 \times 42)^2 + (13 \pm 0) = (9.8 \pm 0) \times 10^3 [\text{sec}] \quad (6.3)$$

$$\{(3.0 \pm 0.1) \times 10^{-5}\} \times (2000 \times 42) + \{(8.9 \pm 0.8) \times 10^{-2}\} = 2.6 \pm 0.1 [\text{sec}] \quad (6.4)$$

906 方法 1 では 1 回の検索に対して約 2.7 時間と見積もられ、生産時には検索機能として運用不可能なシス
907 テムであることがわかる。方法 2 では終了時点においても数秒で処理を終えることができるため、生産を
908 通して十分に使うことができると考えられる。

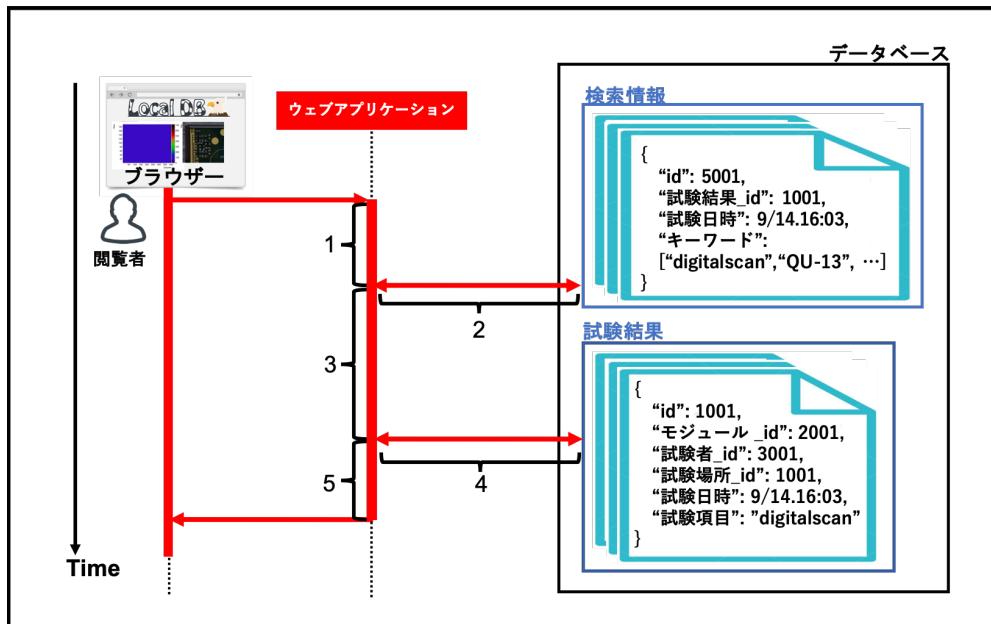


図 6.8 方法 2 の検索における詳細処理。図のように全体の流れにおけるアプリケーション内部での各処理、データベースから情報取得の各処理に 1 から 5 の番号をつけそれぞれにかかる処理時間を測定した。

6.3 改善方法の処理時間測定

より処理時間を短くすることを目的として、新たな検索処理アルゴリズムの考案と測定を行った。詳細について以下に示す。

6.3.1 方法 2 における検索処理時間の詳細調査

先述したように、方法 2 では処理時間が改善した。この方法 2 について、処理時間の詳細を知るために追加で測定を行った。アプリケーション層での各処理について、以下のように番号を付ける。

1. キーワードを受け取り、検索情報コレクションに検索をかけるまでの処理。
2. 検索情報コレクションに検索をかけ、情報を受け取る処理。
3. ドキュメントを受け取り該当する試験結果 ID をまとめ、試験結果に対して検索をかけるまでの処理。
4. 試験結果コレクションに検索をかけ、情報を受け取る処理。
5. ドキュメントを受け取りデータを整形、ブラウザにレスポンスを返すまでの処理。

イメージを図 6.8 に示す。

ボトルネックとなっている処理を測定するために、上述した各処理にかかる時間の測定を行った。測定は試験結果数が 10,000 の場合に行なった。また測定は 20 回行った。

結果を図 6.9 に示す。

図より処理 3、5 の割合が大きいことがわかる。これらの処理について、特に以下の処理の割合が大き

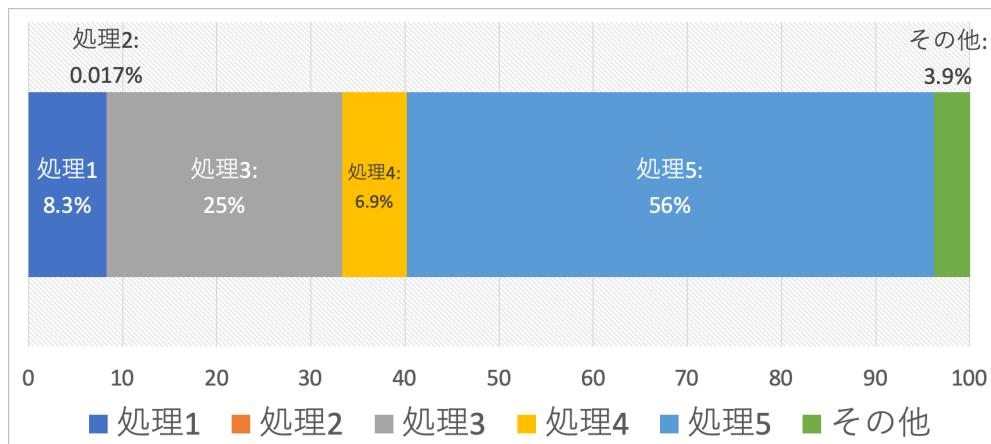


図 6.9 方法 2 における詳細処理時間の測定結果。図 6.8 における 5 つの詳細処理にかかる時間の割合を表している。処理 3、5 にあたるコレクション検索実行後のアプリケーション内での処理に多く時間がかかっていることが分かる。

表 6.3 処理 3,5 における型変換処理 6.5 の割合。処理 3、5 について、表の割合より、型変換処理 6.5 が支配的であることが分かる。

処理	全体 [sec]	処理 6.5[sec]	割合 [%]
3	0.091 ± 0.011	0.089 ± 0.005	97 ± 0
5	0.21 ± 0.00	0.18 ± 0.00	86 ± 1

926 いことがわかった。

取得した複数ドキュメントから Python リストへの型変換 (6.5)

927 図 6.8 における処理 3、5 において、型変換処理 6.5 の割合を表 6.3 まとめた
 928 この変換処理について、あるコレクションにおける全ドキュメント数に対する Python リスト変換処理
 929 時間の関係を測定した。結果を図 6.10 に示す。全ドキュメント数に対して線形性を示していることがわ
 930 かる。方法 2 の検索処理については処理 6.5 が支配的であることが分かった。

931 6.3.2 改善点

932 測定を踏まえ、改善方法として以下の項目を検討した。ここでは、上述したように使用しているデータ
 933 構造やフレームワークの変更はせずに処理時間を改善することを前提としている。

- 934 ● コレクション検索処理の回数を減らす。
- 935 ● 検索対象コレクションのドキュメント数を減らす。

936 上述した 2 つを目的として、以下の 2 つの方法を新しく考え処理時間測定を行った。

- 937 3. 検索情報のコレクションに一覧表示に必要な情報を保持、参照。
- 938 4. 方法 3 に付け加えて、検索情報のドキュメントを複数コレクションに分散、マルチスレッドを用い
 939 た検索処理の並列化。

940 方法 3 については一覧表示に必要な情報を検索情報のドキュメントが持つことで、データベースに対す

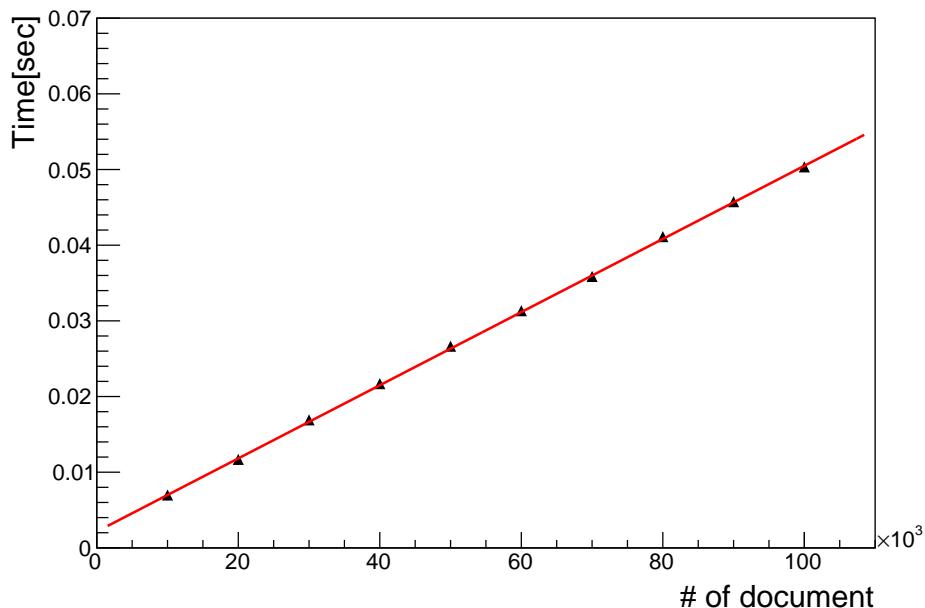


図 6.10 ドキュメント数に対する型変換処理時間の関係。コレクション検索後の型変換に要する処理時間は、図のようにドキュメント数に対して線形となっていることが分かる。

941 る検索の回数を減らすことを目的としている。方法 4 については方法 3 の検索処理の数を減らすことに加
942 えて、ドキュメントの数を分散し並列処理をすることで処理時間の改善を図っている。イメージをそれぞ
943 れ図 6.11、6.12 に示す。

944 方法 3、4 について、章 6.2 と同じ内容の測定を行った。方法 2 のものと合わせた結果を 6.13 に示す。
945 方法 4 について、分散するコレクション数は 10 個、スレッド数には 2 とした。方法 2 に比べて、方法 3、
946 4 共に処理時間が改善していることがわかる。

947 方法 3、4 を比べると傾きに差が見られる。そのため、方法 4 はドキュメント数が多くなった時に有効
948 であると考えられる。方法 4 に関しては今回はコレクション数を 10、スレッド数を 2 としたが、それぞ
949 れ最適な数を検討することで更なる改善ができる可能性がある。

950 得られた方法 3,4 に関する関係を式 6.7、6.8 に示す。

$$\begin{aligned}
 y &= ax + b \\
 a &= (2.9 \pm 0.1) \times 10^{-5} \\
 b &= (5.0 \pm 1.1) \times 10^{-3}
 \end{aligned} \tag{6.6}$$

$$\begin{aligned}
 y &= ax + b \\
 a &= (2.7 \pm 0.1) \times 10^{-5} \\
 b &= (2.2 \pm 1.0) \times 10^{-2}
 \end{aligned} \tag{6.7}$$

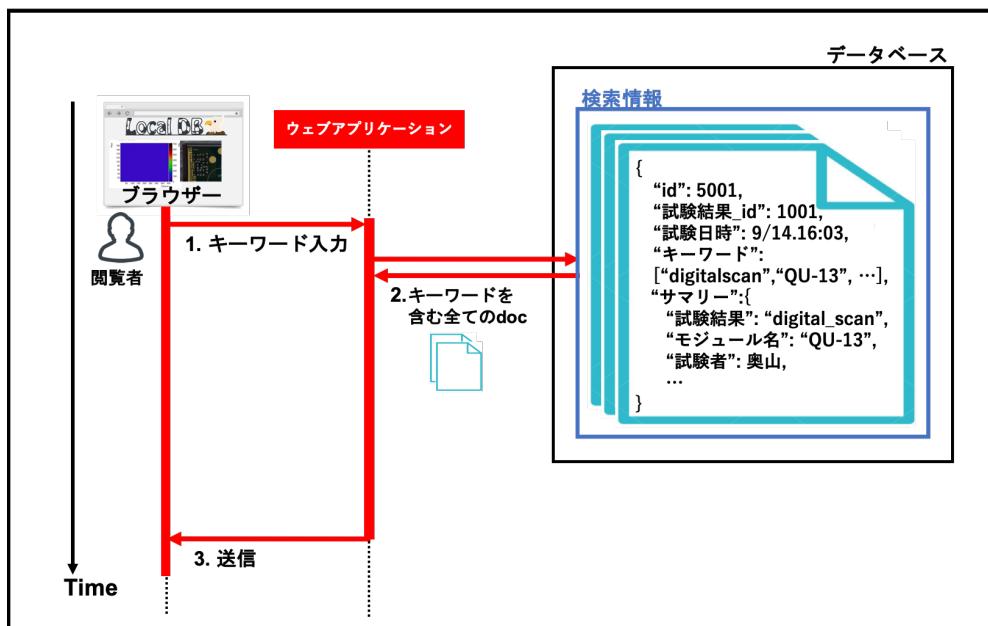


図 6.11 検索機能実装方法 3:検索情報と共に一覧表示に必要な情報を保持、参照。方法 2 では検索情報コレクションより、条件に一致する試験結果 ID を取得し、実際の試験結果に対して再度検索をかける流れとなっていたが、この方法では一覧表示に必要な情報も全て検索情報のコレクション内で保持する。こうすることで検索機能において必要な情報の全てが 1 つのコレクションにまとまり、コレクション検索の処理が 1 回で済むため、処理時間が改善すると考えられる。

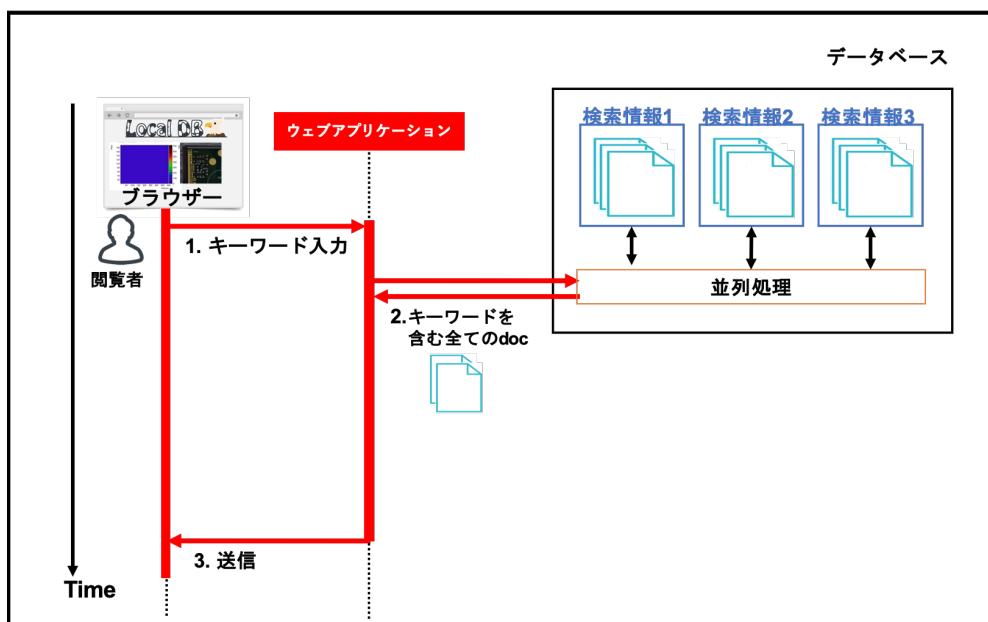


図 6.12 検索機能実装方法 4:検索情報コレクションを分散、マルチスレッドを使用。方法 3 に加えて、検索情報コレクションを分散し、並列処理を行うことで、1 つのコレクションあたりに含まれるドキュメント数を減らし、コレクション検索にかかる時間を削減できると考えられる。

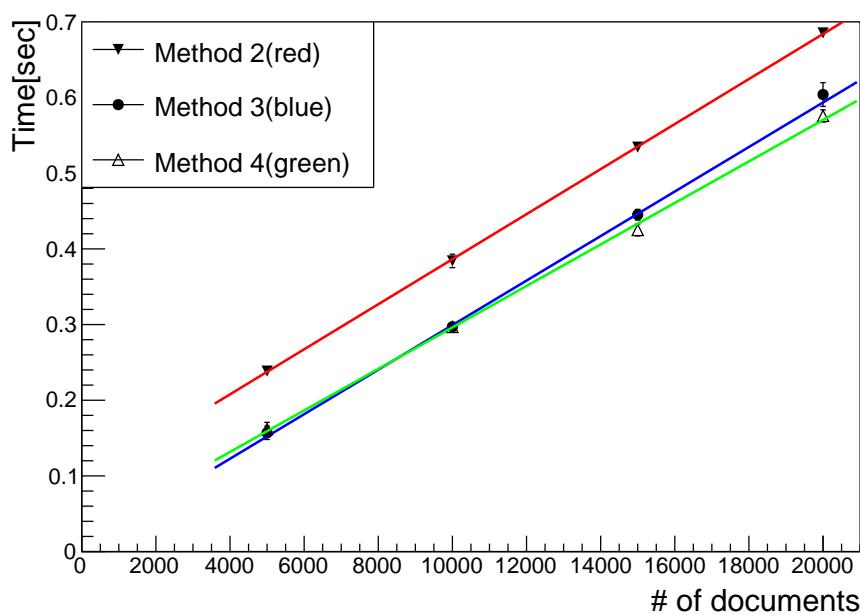


図 6.13 方法 3、4 に対する処理時間測定結果。横軸が試験結果のドキュメント数、縦軸が処理時間を表している。赤、青、緑がそれぞれ方法 2、3、4 を用いたものである。方法 2 に比べて 3、4 共に改善していることが分かる。方法 3、4 を比べると傾きが小さくなっていることが分かる。これよりドキュメント数が大きい場合に方法 4 は有効である。

951 第 7 章

952 中央データベースとローカルデータベースの同期

953

954 この章では中央データベースとローカルデータベースのデータ同期機能に関する調査と各ツールにおける改善策の考案、処理時間測定を行った。詳細について以下で説明する。

955 7.1 サーバーの設置場所による処理時間の違い

956 4 章で述べたように、中央データベースはチェコに設置されている。そのため試験結果のアップロード
957 に関する、各組み立て機関から接続しデータ送信する処理時間は、機関の場所に大きく依存すると考えら
958 れる。世界的にデータ同期ツールが不自由なく動くことに向か開発、改善に役立てる目的とし
959 て、データベース間の情報通信にかかる処理時間を、以下の 3 つの場所に置かれているサーバーを用いて
960 測定した。組み立て機関及びローカルデータベースの設置場所はヨーロッパ、アメリカ、日本の 3 つの地
961 域に分布している（付録 C）。それぞれにおける代表機関として以下の 3 つに設置されたサーバーを用い
962 て調査を行った。

- 963 • 日本、高エネルギー加速器研究所 (KEK)
- 964 • アメリカ、バークレー研究所 (LBL)
- 965 • スイス、欧州原子核研究機構 (CERN)

966 各サーバーの性能を表 7.1 に示す。また各サーバーが置かれている場所の位置関係を図 7.1 に示す。
967 これらのサーバーは実際に生産の際に使用するものと同程度の性能を持ち、サーバーが置かれている
968 ネットワーク環境も生産時と同じであると仮定している。

表 7.1 各ローカルデータベースサーバーの性能一覧。今回の調査に利用したサーバーの性能を示す。

KEK(日本)、LBL(アメリカ)、CERN(スイス)に設置されたサーバーを用いた。

設置機関	CPU Type	Core	Thread	Clock speed[GHz]	Memory [GB]	Disk [GB]
KEK(日本)	Intel(R) Core(TM) i7-9700K	8	16	3.6	32.66	1800 + 1800
LBL(アメリカ)	Intel(R) Core(TM) i7-8700	6	12	3.7	32.63	233
CERN(スイス)	Intel(R) Core(TM) i7-4790	4	8	3.6	32.69	238.5 + 3700 + 3700



図 7.1 各サーバーの設置場所。赤点で示しているのがそれぞれローカルデータベースサーバーの設置位置であり、オレンジで示しているのが中央データベースである。距離としては CERN が一番近く、LBL、KEK の順番となっている。

表 7.2 同期ツールの中で使用する中央データベースの主な API 一覧。同期ツールにおいて、中央データベースの情報取得には提供されているいくつかの API を用いており代表的なものをいかに示す。この API を Python を用いて実行することで、情報取得や試験結果のアップロードをすることができる。

関数名	処理の内容	本ツールでの使用用途
getComponent	登録した部品情報の取得	主にダウンロード時におけるモジュールやチップの情報取得に用いる。
listComponents	登録した部品情報一覧の取得	主にダウンロード時におけるモジュール情報一覧取得に用いる。
uploadTestRunResults	テスト結果生成	読み出し試験結果生成の際に用いる。
createTestRunAttachment	あるテスト結果に対するバイナリファイルの添付	読み出し試験結果生成後にファイルを添付する際に用いる。

970 7.1.1 同期ツールに使用する API

971 中央データベースとの同期には、中央データベースで開発された API を使用している。ローカルデータベースとの同期ツールの中で使用している主な API を表 7.2 に示す。

973 7.1.2 API 使用にかかる時間

974 上述した API 使用時の処理時間を各サーバーで測定した。以下の 3 つの測定を行なった。

- 975 • getComponent を用いた、登録モジュール情報 1 つの取得時間測定。
- 976 • createTestRunAttachment を用いて、ある試験結果ページに 1Byte のデータファイルを添付する時間測定。
- 977 • createTestRunAttachment を用いて、ある試験結果ページに容量の異なるデータファイルを添付、容量に対する時間依存性を測定。

980 最初の 2 項目に関して、まとめたものを表 7.3 に示す。ファイル容量と処理時間の関係を図 7.2 に示す
981 ここで、どの場合においても KEK における処理時間が最も長いことがわかる。そのため、KEK における
982 処理時間を測定し、ツールの開発、改善について考えることとした。(1 週間でのアクセススピードの変
983 位を見るために、1 週間通してのウェブページアクセス時間のモニタリングを行っている。ここか付録に

表 7.3 中央データベース API 実行時の処理時間測定結果。左の結果は表 7.2 における”getComponent”を用いてモジュール 1 つの情報を取得するのにかかった時間、右は”createTestRunAttachment”を用いて 1Byte のファイル送信にかかった時間である。どのサーバーにおいても 0.3 秒以上の処理時間がかかっていることがわかり、データベースへの接続、情報取得にかかる時間が読み取れる。3 つのサーバーを比べると、どちらの場合も KEK サーバーでの処理時間が一番大きいことが分かる。

サーバー	処理時間 [秒]	サーバー	処理時間 [秒]
KEK	0.47 ± 0.01	KEK	0.83 ± 0.01
LBL	0.37 ± 0.02	LBL	0.34 ± 0.03
CERN	0.28 ± 0.01	CERN	0.48 ± 0.03

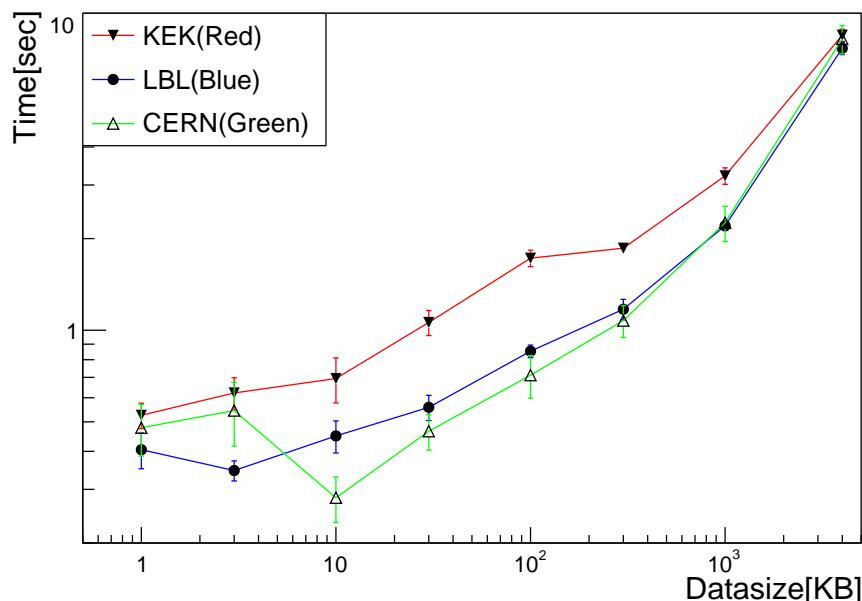


図 7.2 ”createTestRunAttachment”を用いた添付処理におけるファイル容量と処理時間の関係。それぞれのサーバーで 1、3、10、30、100、300KB、1、4MB のファイル送信にかかる時間を測定した。どの点においても KEK サーバーが最も処理時間を要していることが分かる。またこのグラフについての詳しい考察を付録 E に記す。

表 7.4 ダウンロード機能を用いて保存する情報一覧。ダウンロード機能を用いて中央データベースからローカルデータベースに保存する情報の一覧を示している。保存の際にはモジュール、FE チップにそれぞれ分かれたドキュメントに保存される。

部品	情報
モジュール	シリアルナンバー
	搭載 FE チップの種類
	登録機関
	搭載 FE チップの枚数
FE チップ	シリアルナンバー
	FE チップ ID(モジュール上の位置を表す情報)
	登録機関

985 7.2 モジュール情報のダウンロード

986 7.2.1 ダウンロードする情報と構造

987 中央データベースから、モジュール及び FE チップの情報をダウンロードする機能を開発、実装した。

988 ダウンロードする情報の詳細について表 7.4 に示す。

989 ダウンロードされたモジュール、FE チップのドキュメントの例をリスト 7.1、7.2 に示す。

ソースコード 7.1 ダウンロードしたモジュール情報のドキュメントの例。ドキュメントが表 7.4 の情報を持つことが分かる。

```

990
991 1 {
992 2     "_id" : ObjectId("5fa79114e615fa000a1a5976"),
993 3     "name" : "20UPGR00000001",
994 4     "chipType" : "RD53A",
995 5     "serialNumber" : "20UPGR00000001",
996 6     "chipId" : -1,
997 7     "componentType" : "module",
998 8     "address" : "5fd597fdf7339bbf26b87fb2",
999 9     "children" : 1,
1000 10    "sys" : {
1001 11        "mts" : ISODate("2020-12-13T04:26:37.989Z"),
1002 12        "cts" : ISODate("2020-12-13T04:26:37.989Z"),
1003 13        "rev" : 0
1004 14    },
1005 15    "dbVersion" : 1.01,
1006 16    "user_id" : -1,
1007 17    "proDB" : true
1008 18 }
```

ソースコード 7.2 ダウンロードした FE チップ情報のドキュメントの例。ドキュメントが表 7.4 の情報を持つことが分かる。

```

1010
1011 1 {
1012 2     "_id" : ObjectId("5fa79560e615fa000a1a5a16"),
1013 3     "name" : "20UPGFC9999999",
1014 4     "chipType" : "RD53A",
1015 5     "serialNumber" : "20UPGFC9999999",
1016 6     "chipId" : 0,
1017 7     "componentType" : "front-end_chip",
1018 8     "address" : "5fd597fdf7339bbf26b87fb2",
1019 9     "children" : -1,
1020 10    "sys" : {
1021 11        "mts" : ISODate("2020-12-13T04:26:37.984Z"),
1022 12        "cts" : ISODate("2020-12-13T04:26:37.984Z"),
1023 13        "rev" : 0
1024 14    },
1025 15    "dbVersion" : 1.01,
```

表 7.5 登録したモジュールのダウンロード処理時間測定結果。登録したそれぞれのモジュールについてダウンロードにかかる時間を測定した。表より 1 つあたり平均 4 秒の時間がかかっていることが分かる。

モジュール	処理時間
20UPGM20030004	3.8
20UPGM20030001	3.7
20UPGM20030003	5.9
20UPGM20030006	3.6
20UPGM20030022	3.8
20UPGM20030024	3.3
平均	4.0 ± 0.4

```
1026   16      "user_id" : -1,
1027   17      "proDB" : true
1028 }
```

1030 7.2.2 処理の流れ

1031 ダウンロード機能における処理の流れのイメージを図 7.3 に示す。

1032 7.2.3 機能確認

1033 KEK で組み立てられた 6 台の Quad モジュールを中央データベースに登録し、ダウンロードを行つ
1034 た。登録したモジュールを表 7.4、ダウンロードをしてアプリケーションで確認した様子を図 7.5 に示す。

1035 7.2.4 処理時間測定

1036 ダウンロードの処理時間を測定した。これについてまとめたものを表 7.5 に示す。Quad モジュール 1
1037 つに対して平均して 4.0 ± 0.4 秒の処理時間がかかっている。

1038 7.2.5 処理時間詳細

1039 ダウンロード処理時間の改善に向けて、処理時間の詳細について以下の測定した。

- 1040 1. 中央データベースからモジュール情報の取得.
- 1041 2. データベースでの FE チップ確認処理.
- 1042 3. 中央データベースからバアモジュール情報の取得.
- 1043 4. 中央データベースから FE チップ情報の取得 (4 枚分).
- 1044 5. ローカルデータベースへの情報の書き込み (モジュール、FE チップ、品質試験情報).

1045 情報取得のイメージを表 7.6 に示す。このように Quad モジュールの場合、ダウンロードの流れの中で
1046 合計して 7 回、データベース API を用いて情報取得を行う。
1047 結果を表 7.6 に示す。

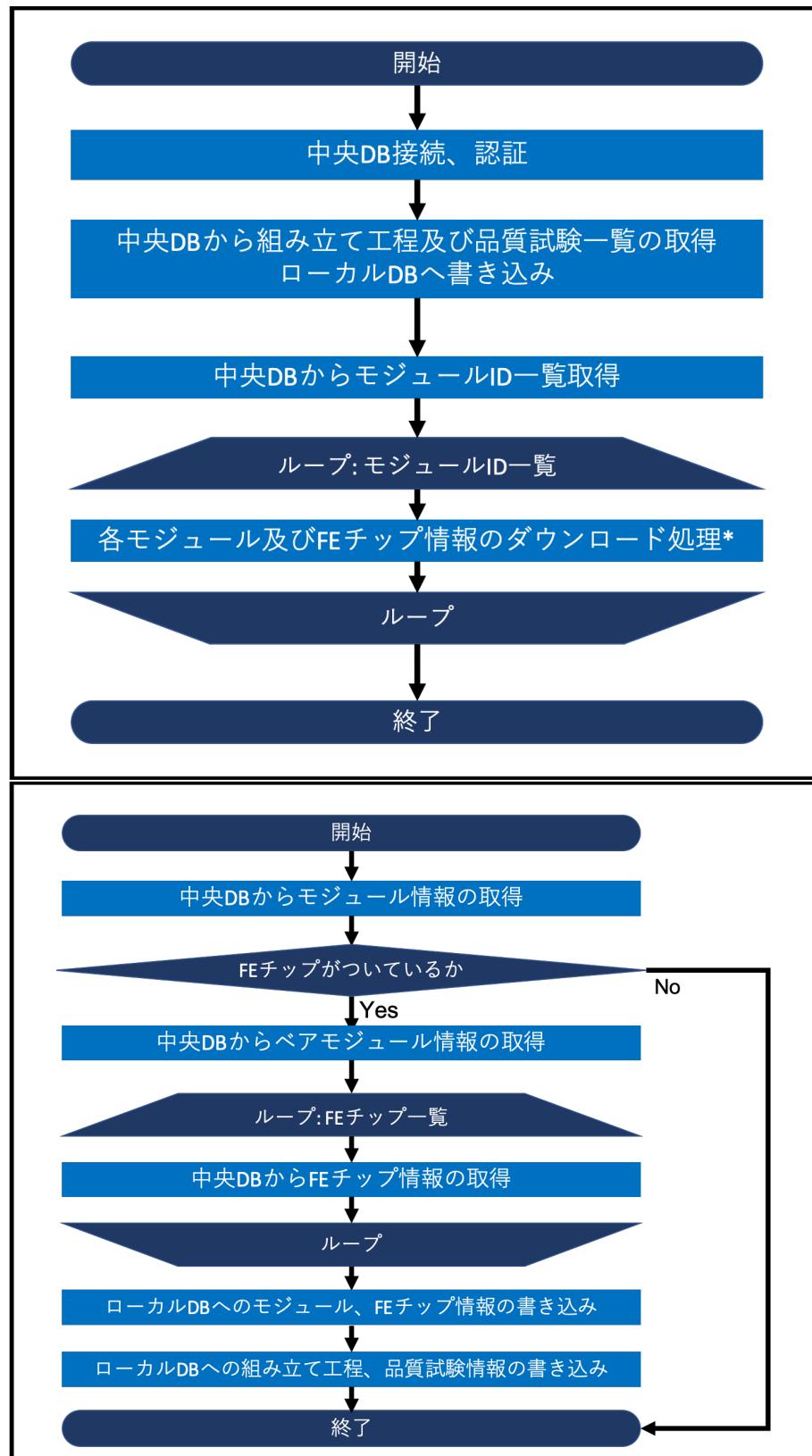


図 7.3 ダウンロード処理における流れのイメージ図。上図が処理全体の流れを表すものであり、下図は各モジュール情報のダウンロードにおける処理の流れを示している。上図中のループ構造の 1 処理が下図に対応している。流れの中で複数回中央データベースに接続し、モジュールや FE チップの情報取得をしていることが分かる。

Module	Bare Module	Sensor	PCB	Carrier	FE chip
20UPGM20030004	20UPGB40500019	20UPGS83300002	20UPGPQ0030004	20UPGMC0210000	20UPGFC0014659 20UPGFC0014658 20UPGFC0014675 20UPGFC0014691 20UPGFC0014644 20UPGFC0014628 20UPGFC0014660 20UPGFC0014708 20UPGFC0014677 20UPGFC0014629 20UPGFC0014693 20UPGFC0014646 20UPGFC0016282 20UPGFC0016281 20UPGFC0016279 20UPGFC0016280 20UPGFC0016278 20UPGFC0016267 20UPGFC0016235 20UPGFC0016242 20UPGFC0016276 20UPGFC0016266 20UPGFC0016220 20UPGFC0016227
20UPGM20030001	20UPGB40500020	20UPGS83300003	20UPGPQ0030001	20UPGMC0210001	
20UPGM20030003	20UPGB40500021	20UPGS83300004	20UPGPQ0030003	20UPGMC0210002	
20UPGM20030006	20UPGB40500022	20UPGS83300001	20UPGPQ0030006	20UPGMC0210003	
20UPGM20030022	20UPGB40500023	20UPGS83300005	20UPGPQ0030022	20UPGMC0210004	
20UPGM20030024	20UPGB40500024	20UPGS83300006	20UPGPQ0030024	20UPGMC0210005	

図 7.4 登録した Quad モジュールとその構成部品のシリアルナンバー一覧。左から、登録したモジュール、搭載されているベアモジュール、シリコンセンサー、PCB、モジュールキャリア、FE チップの中央データベース内でのシリアルナンバーを示している。Quad モジュールであるため、FE チップをそれぞれ 4 つ搭載している。

表 7.6 ダウンロード機能における詳細処理にかかる時間測定。図 7.6 より、中央データベースに接続、情報取得を合計して 7 回行っており、それが処理 1 から 4 に対応する。どの処理においても 0.5 秒程度の時間がかかっていることが分かる。処理 5 はローカルデータベースへの書き込み処理であるが、他の処理に比べて十分に小さいことが分かる。

処理	時間
1	0.60 ± 0.07
2	0.55 ± 0.07
3	0.61 ± 0.04
4	0.46 ± 0.04 0.71 ± 0.19 0.51 ± 0.04 0.57 ± 0.12
5	0.0025 ± 0.0011
合計	4.0 ± 0.1

1. 中央データベースからモジュール情報の取得.
2. データベースでの FE チップ確認処理.
3. 中央データベースからベアモジュール情報の取得.
4. 中央データベースから FE チップ情報の取得 (4 枚分).
5. ローカルデータベースへの情報の書き込み (モジュール、FE チップ、品質試験情報).

ProdDB web

IN PROGRESS	Module - Outer system quad module	20UPGM20030001
IN PROGRESS	Module - Outer system quad module	20UPGM20030024
IN PROGRESS	Module - Outer system quad module	20UPGM20030022
IN PROGRESS	Module - Outer system quad module	20UPGM20030006
IN PROGRESS	Module - Outer system quad module	20UPGM20030003
IN PROGRESS	Module - Outer system quad module	20UPGM20030004

 Download

LocalDBX TOP / COMPONENTS / TEST

ITk database for Yarr Component List

Input keywords Partial match Perfect match

RD53A (11 modules)

Module Name	Chip Name	Latest Result						Tag
		Current Stage	Test Type	User	Site	Date	Link	
20UPGR90020026	20UPGFC0028965 20UPGFC0028950 20UPGFC0028951 20UPGFC0028952	None	None	None	None	None		
20UPGR90000004	20UPGFC0008036 20UPGFC0007994 20UPGFC0007972 20UPGFC0008035	None	None	None	None	None		
20UPGR30000001	20UPGR33000000 20UPGR33000001 20UPGR33000002 20UPGR33000003	None	None	None	None	None		
20UPGR10099999	20UPGFC999995 20UPGFC999996 20UPGFC999997 20UPGFC999998	None	None	None	None	None		
20UPGR00000001	20UPGFC999999	None	None	None	None	None		
20UPGM20030024	20UPGFC0016276 20UPGFC0016266 20UPGFC0016220 20UPGFC0016227	None	None	None	None	None		
20UPGM20030022	20UPGFC0016278 20UPGFC0016267 20UPGFC0016235 20UPGFC0016242	None	None	None	None	None		
20UPGM20030006	20UPGFC0016282 20UPGFC0016281 20UPGFC0016279 20UPGFC0016280	None	None	None	None	None		
20UPGM20030004	20UPGFC0014659 20UPGFC0014658 20UPGFC0014675 20UPGFC0014691	None	None	None	None	None		
20UPGM20030003	20UPGFC0014677 20UPGFC0014629 20UPGFC0014693 20UPGFC0014646	None	None	None	None	None		
20UPGM20030001	20UPGFC0014644 20UPGFC0014628 20UPGFC0014660 20UPGFC0014708	None	None	None	None	None		

図 7.5 登録した Quad モジュールのダウンロードの様子。上図が中央データベースのウェブページを表しており、下図がローカルデータベースのものである。上図で登録したモジュール一覧を確認でき、赤枠で囲っているところでシリアルナンバーを見ることができる。ダウンロード実行後は下図のようにローカルデータベースで対応するモジュールを確認することができる。ローカルデータベースではモジュール情報に加えて FE チップの情報も取得するため、下図の表ではこれらのシリアルナンバーも確認できる。

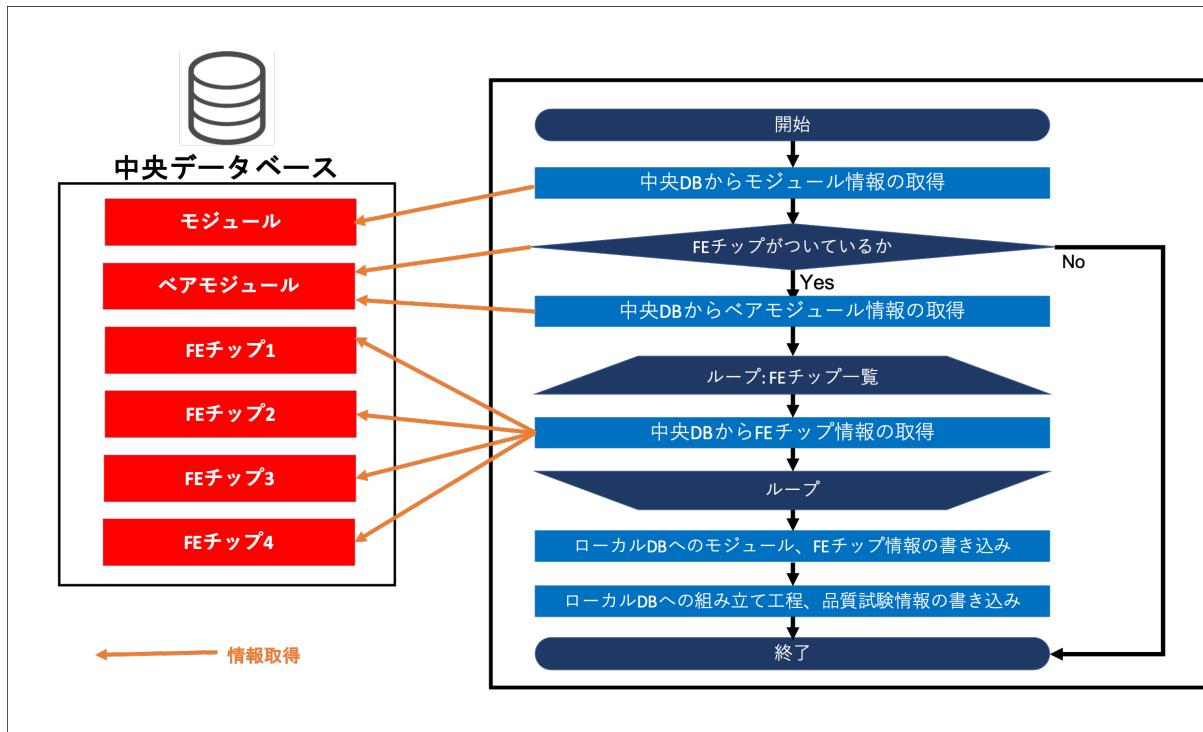


図 7.6 モジュール及び構成部品情報取得のイメージ。図のように処理の流れの中で合計 7 回中央データベースに接続し、モジュール、ペアモジュール、FE チップの情報取得を行っている。

1048 この結果より、各構成部品情報の取得（モジュール、ペアモジュール、FE チップ）の取得にそれぞれ均
1049 等に処理時間がかかることがわかった。

1050 7.2.6 生産時における見積もり

1051 現在ダウンロード機能のオプションとして、以下の 2 つを実装している。

- 1052 1. モジュール 1 つをダウンロードする機能。
1053 2. 登録されている全てのモジュールの一括ダウンロード機能。

1054 オプション 1 の見積もり値は、表 7.5 の平均値として $4.0 \pm 0.4[\text{sec}]$ となる。オプション 2 の見積もり
1055 値は、生産時には最大で 10,000 台のモジュールが中央データベースに登録されることから、以下のよう
1056 になる。

$$(4.0 \pm 0.4)[\text{sec}] \times 10,000 = 11 \pm 1[\text{hour}] \quad (7.1)$$

1057 各機関で各モジュールの組み立てを始める際に、中央データベースへのモジュール登録及びローカル
1058 データベースへのダウンロードを行うことを想定している。これは 1 つずつ行うため、このような場合は
1059 オプション 1 を用いる。

1060 オプション 2 の使用例として、モジュールの組み立て工程の途中で、複数のモジュール ($O(100)$ 程度)
1061 を機関 1 から機関 2 に輸送する場合をあげる。機関 2 では機関 1 で登録された全てのモジュール情報を
1062 ダウンロードしてくる必要があるため、このオプションを使用する。輸送に要する時間として 1 日と仮定
1063 すると、この時間を利用してダウンロード処理を行う必要がある。

1064 7.2.7 改善点の考案と見積もり

1065 オプション 2 について、11 時間の処理時間をする見積もりとなっている。より円滑な同期処理に向
1066 けて、改善方法について考える。

1067 一括ダウンロード機能については以下の改善点が考えられる。

- 1068 1. モジュールの現在位置に対応したものののみのダウンロード.
- 1069 2. FE チップの登録機関を取得しない.
- 1070 3. モジュールのプロパティとして、ダウンロードに必要な情報を全て保存.
- 1071 4. データベース API を改良し、モジュール一覧取得の際に構成要素の情報を取得できるようにする.

1072 これらについて詳細と処理時間の見積もりを以下で行う。

1073 改善案 1: モジュールの現在位置に対応したものののみのダウンロード

1074 上述した機関が途中で変更となるような組み立ての流れにおいて、全てのモジュール ID をダウンロー
1075 ドする必要はない。中央データベースにはモジュールの現在位置情報を保持しているため、機能実行者と
1076 位置が同じもののみをダウンロードするアルゴリズムにすれば処理時間を改善できる。見積もりとして
1077 は、ダウンロード対象となるモジュール数を n とすると、以下のようになる。

$$(11 \pm 1) \times \frac{n}{10000} [\text{hour}] \quad (7.2)$$

1078 改善案 2: FE チップの登録機関を取得しない

1079 表 7.6 よりダウンロードの際に、FE チップの情報取得を行っている。これは FE チップ登録機関の情
1080 報を取得しローカルデータベースに保存するためであるが、登録機関の情報は組み立て現場で扱う作業と
1081 しては、必要な情報ではない。そのため、現段階では FE チップのデータ取得処理は割愛することができる。
1082 これにかかる処理時間は表 7.6 より、合計して $2.3 \pm 0.2[\text{sec}]$ となるため、その場合オプション 2 の
1083 処理時間の見積もりは、以下のようになる。

$$\{(4.0 \pm 0.4) - (2.3 \pm 0.2)\} [\text{sec}] \times 10,000 = 4.9 \pm 0.8 [\text{hour}] \quad (7.3)$$

1084 この改善策のデメリットとしては、FE チップの情報取得処理を省くとローカルデータベースで扱いたい
1085 情報が将来的にできた場合に保存できないことである。例えば各 FE チップの最適動作電圧のようにモ
1086 ジュール読み出しに対して有益な情報は保存し、迅速に確認したいという方針になる可能性もある。

1087 改善案 3: モジュールのプロパティとして、ダウンロードに必要な情報を全て保存

1088 モジュールのプロパティとして、FE チップの名前等のダウンロードに必要な情報を書いておくと、表
1089 7.2 における listComponents によるモジュール一覧取得でその情報を参照することができる。こうする
1090 ことで、表 7.6 において、ペアモジュールや FE チップの情報取得を省くことができる。この場合処理時
1091 間は、合計して $2.9 \pm 0.2[\text{sec}]$ となるため、その場合オプション 2 の処理時間の見積もりは、

$$\{(4.0 \pm 0.4) - (2.9 \pm 0.2)\} [\text{sec}] \times 10,000 = 3.1 \pm 0.8 [\text{hour}] \quad (7.4)$$

1092 これのデメリットは、データベースの中でデータが冗長になってしまうことである。FE チップの名前

1093 情報がモジュールのプロパティにも保存されると、データベース内部で冗長性を持ってしまい、編集
1094 が加えられた場合などこれを管理するのが難しくなる。

1095 改善案 4:データベース API を改良し、モジュール一覧取得の際に構成要素の情報を取得できるようにする

1096 現在、表 7.2 の listComponents を用いた時にはモジュール一覧の情報は取得できるが、各モジュール
1097 に対して構成要素は取得できない。そのため表 7.6 のようにモジュールごとに中央データベースに接続
1098 し、部品情報を取得している。ダウンロードに必要な情報を listComponents で一括で取得できるような
1099 仕様に API の変更を行えば、中央データベースへの接続は一回ですみ、処理時間を削減できると考えら
1100 れる。この場合、中央データベースの内部構造を知り、一括で取得しデータ送信をする場合にどれだけの
1101 時間を要するかを見積もり、今の場合と比較する必要がある。

1102

1103 現段階では組み立ての試験段階であり、現場で必要な情報、世界各地での組み立て工程の流れ等を検討
1104 している段階である。ここで述べたような改善策を組み合わせ、変更を加えていく必要がある。また中央
1105 データベースとの同期におけるネットワーク速度は地理的な距離により異なるため、この点も考慮に入れ
1106 る必要がある。今回の測定は日本での測定であるため、他の国においては処理速度は改善すると考えられる。

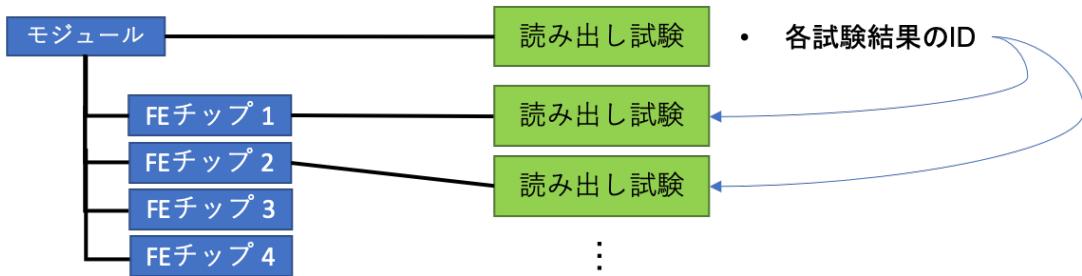


図 7.7 中央データベースにおける読み出し試験結果の構造。YARR の出力ファイル及びローカルデータベースのデータ構造において、読み出し試験結果は全て FE チップに紐つけられている。そのため、図のように中央データベースにおいてもこのデータ構造を保持する形でアップロードを行う。モジュールの結果は各 FE チップの試験結果に対する ID を持つことで紐付けを行っている。

表 7.7 中央データベースにおける読み出し試験結果に関する情報一覧。モジュール及び FE チップが中央データベース内で持つ試験結果の情報を示している。

部品	試験情報、結果	添付ファイル
モジュール	モジュール環境温度 FE チップにつく読み出し試験結果の ID	
FE チップ	ピクセル解析結果	試験結果データファイル 読み出し設定ファイル その他設定ファイル

1107 7.3 読み出し試験結果のアップロード

1108 4 章で述べたように、読み出し試験結果について中央データベースへアップロードするツールを開発し
1109 た。以下で詳細を述べる。

1110 7.3.1 アップロードする情報とその構造

1111 読み出し試験結果について、中央データベースにアップロードする情報を以下に記す。

- 1112 • 試験日時.
- 1113 • モジュール周りの環境温度.
- 1114 • ピクセル解析結果.
- 1115 • 各試験結果データファイル.
- 1116 • 読み出し設定ファイル.
- 1117 • その他設定ファイル (DB、ユーザ、組み立て機関等).

1118 中央データベースにおける読み出し試験の構造に関して、YARR を用いて行った読み出し結果は全て
1119 FE チップ毎に取得、保存される。そのため、データベースの内部でも FE チップに読み出し試験結果を
1120 紐づける構造を設け、モジュールの結果では各 FE チップの結果ページの ID を持つ構造とした。イメー
1121 ジを図 7.7 に示す。

1122 中央データベースにおいてモジュール、FE チップの試験結果が持つ情報を表 7.7 にまとめた。

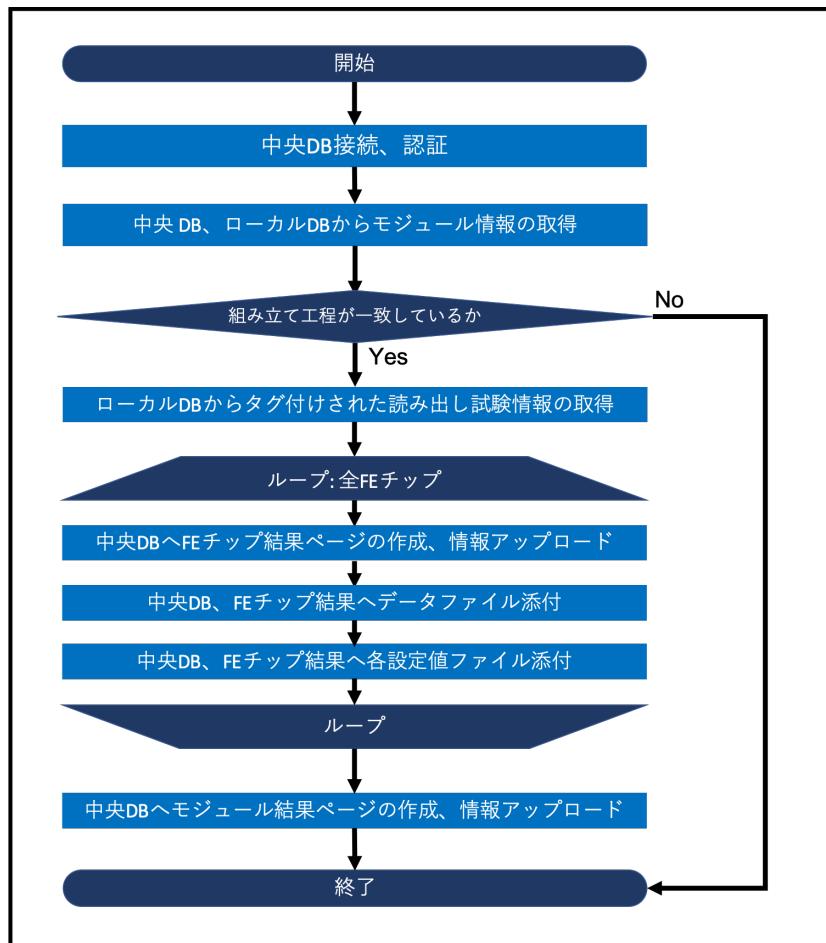


図 7.8 アップロード処理に関する流れのイメージ図。流れの中には FE チップに関するループ構造があり、ここで FE チップの結果生成、結果ファイルのアップロードを行う。最後にモジュールに対する結果生成とアップロードを行う。

1123 7.3.2 処理の流れ

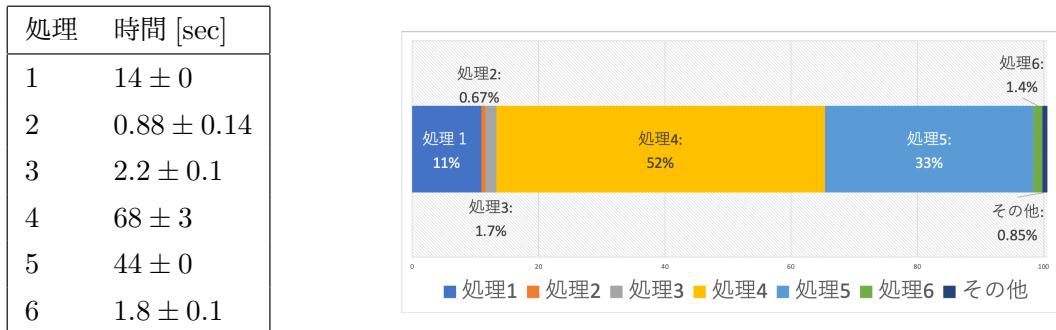
1124 アップロード機能における処理の流れのイメージを図 7.8 に示す。流れの中には共通して行われる処理
1125 と、各 FE チップに対して行われる処理がある。

1126 7.3.3 処理時間測定

1127 上述した処理のツールを開発し、処理時間測定を行った。ここで行ったアップロードする試験項目は章
1128 5 で行ったデモンストレーションのものと同じで以下の項目とする。

- 1129 • デジタル回路読み出し
- 1130 • アナログ回路読み出し
- 1131 • Threshold 測定
- 1132 • ToT 測定
- 1133 • ノイズ占有率測定

1134 これらの項目についてアップロードを行い、その処理時間を測定した。KEK のサーバーを用いてアッ



1. 中央データベース接続、認証。
2. 中央データベース、ローカルデータベースからモジュール情報の取得。
3. 中央データベースに FE チップ結果ページの作成。結果情報をアップロード。
4. 2 で作成した結果に対して、各データファイルを添付。
5. 2 で作成した結果に対して、各設定値ファイルを添付。
6. 中央データベースにモジュールの結果ページの作成、結果情報をアップロード。

図 7.9 アップロード機能における詳細処理測定結果。アップロード機能において、各詳細処理時間を測定した結果である。左図は測定値であり、右図はそれぞれの割合を示したものである。右図より、処理 4、5 の結果ファイルの添付、設定値ファイルの添付に多く時間がかかっていることが分かる。

1135 プロード処理を 20 回を行い、5 項目全体でかかる時間を測定した。以下のようにになった。

$$(1.3 \pm 0.0) \times 10^2 [\text{sec}] \quad (7.5)$$

1136 ここで処理流れの表 7.8 より特に以下の詳細処理を抜粋し、それぞれにかかる時間を測定した。

- 1137 1. 中央データベース接続、認証。
- 1138 2. 中央データベース、ローカルデータベースからモジュール情報の取得。
- 1139 3. 中央データベースに FE チップ結果ページの作成。結果情報をアップロード。
- 1140 4. 2 で作成した結果に対して、各データファイルを添付。
- 1141 5. 2 で作成した結果に対して、各設定値ファイルを添付。
- 1142 6. 中央データベースにモジュールの結果ページの作成、結果情報をアップロード。

1143 結果を表 7.9 に示す。結果データや各設定値のファイル添付に大きく時間がかかっていることが分
1144 かった。

1145 生産時における見積もり

1146 Quad モジュールにおける読み出し試験結果アップロード処理合計時間の見積もりを行った。上述した
1147 測定は SCC であるため FE チップに対する処理は 1 回であるため、Quad モジュールの場合は表 7.9 を
1148 用いて以下のように計算できる。

$$\text{FE チップ処理} : \left\{ (2.2 + 68 + 44) \pm \sqrt{(0.1)^2 + 3^2 + 0^2} \right\} \times 4 = (4.6 \pm 0.1) \times 10^2 [\text{sec}] \quad (7.6)$$

$$(7.7)$$

$$\text{合計} : (4.6 \pm 0.1) \times 10^2 [\text{sec}] + (14 \pm 0) + (0.88 \pm 0.14) + (1.8 \pm 0.1) = 7.9 \pm 0.1 [\text{min}] \quad (7.8)$$

モジュール読み出し試験1回に対して、約8分程度かかる見積もりとなった。円滑なモジュール組み立て、データ管理を行うために同期は速やかに行われることが要求されること、外観検査や平坦性測定のように他の品質試験も同期する必要があることを考慮し、処理時間の改善を試みた。詳細を以下で示す。

7.3.4 改善策

図7.9より処理4、5のファイル添付に大きく要していることが分かる。この現状を踏まえ、各ファイルにおける添付処理時間の測定を行った。測定は上述したものと同様にKEKサーバーを用いて合計20回行った。添付する結果データファイル、設定ファイルの種類とデータ容量、添付処理実行結果、処理時間を表7.8に示す。ここで4MBを超える容量のファイル添付は失敗していることがわかり、アップロード機能の問題点を発見した。

表7.8: アップロード処理における結果、設定値ファイル添付実行結果と処理時間。図7.9より、読み出し試験に対して出力される各ファイルのアップロード実行結果、データ容量と処理時間をまとめた。ここで扱うファイルサイズの合計は94MBである。図よりファイルのデータ容量が大きいほど処理時間が長いことが分かる。std_thresholdscanのようにファイル数が多い項目の場合、合計して大きい処理時間を要することが分かる。全項目において読み出しの設定ファイルにあたるbeforeCfg_chipCfg.json、afterCfg_chipCfg.jsonのアップロードは、中央データベースの容量制限により失敗していることが分かる。(texの技術的にcaptionが複数になってしまふ。)

読み出し項目	ファイル名	実行結果	容量 [KB]	処理時間 [sec]	全体 [sec]
std_digitalscan	EnMask.json	Ok	1,300	3.3 ± 0.1	17±0
	OccupancyMap.json	Ok	1,500	2.9 ± 0.2	
	L1Dist.json	Ok	0.53	0.75 ± 0.12	
	ctrlCfg_ctrlCfg.json	Ok	0.46	0.61 ± 0.08	
	dbCfg_dbCfg.json	Ok	0.60	0.69 ± 0.16	
	siteCfg_siteCfg.json	Ok	0.033	0.61 ± 0.06	
	userCfg_userCfg.json	Ok	0.14	0.66 ± 0.09	
	scanCfg_std_digitalscan.json	Ok	2.2	0.55 ± 0.06	
	beforeCfg_chipCfg.json	Error	7,200	3.0 ± 0.2	
	afterCfg_chipCfg.json	Error	7,200	4.0 ± 0.2	
std_analogscan	EnMask.json	Ok	1,300	3.9 ± 0.1	17±0
	OccupancyMap.json	Ok	1,400	2.6 ± 0.1	
	L1Dist.json	Ok	0.60	0.69 ± 0.16	
	ctrlCfg_ctrlCfg.json	Ok	0.46	0.54 ± 0.05	
	dbCfg_dbCfg.json	Ok	0.60	0.49 ± 0.04	
	siteCfg_siteCfg.json	Ok	0.033	0.48 ± 0.04	
	userCfg_userCfg.json	Ok	0.14	0.58 ± 0.08	
	scanCfg_std_analogscan.json	Ok	2.1	0.45 ± 0.03	
	beforeCfg_chipCfg.json	Error	7,200	2.9 ± 0.2	
	afterCfg_chipCfg.json	Error	7,200	3.9 ± 0.3	
std_thresholdscan	Scurve-30-96.json	Ok	0.98	1.3 ± 0.1	49±1
	Scurve-110-96.json	Ok	0.98	0.45 ± 0.03	
	Scurve-70-96.json	Ok	0.98	0.47 ± 0.04	
	Scurve-150-96.json	Ok	1.0	0.46 ± 0.04	
	Scurve-190-96.json	Ok	1.0	0.64 ± 0.12	
	Scurve-230-96.json	Ok	1.0	0.49 ± 0.03	
	Scurve-270-96.json	Ok	1.0	0.47 ± 0.04	
	Scurve-310-96.json	Ok	1.0	0.47 ± 0.04	
	Scurve-350-96.json	Ok	1.0	0.49 ± 0.03	
	Scurve-390-96.json	Ok	1.0	0.52 ± 0.06	
	Scurve-40-96.json	Ok	1.0	0.46 ± 0.03	
	Scurve-80-96.json	Ok	0.99	0.68 ± 0.13	
	Scurve-120-96.json	Ok	1.0	0.54 ± 0.07	
	Scurve-160-96.json	Ok	1.0	0.51 ± 0.05	
	Scurve-200-96.json	Ok	1.0	0.49 ± 0.04	
	Scurve-240-96.json	Ok	1.0	0.50 ± 0.05	
	Scurve-280-96.json	Ok	1.0	0.48 ± 0.04	
	Scurve-320-96.json	Ok	1.0	0.49 ± 0.05	
	Scurve-360-96.json	Ok	1.0	0.49 ± 0.06	
	Scurve-400-96.json	Ok	1.0	0.45 ± 0.05	
	Scurve-10-96.json	Ok	1.0	0.42 ± 0.03	
	Scurve-50-96.json	Ok	0.99	0.49 ± 0.05	
	Scurve-90-96.json	Ok	0.99	0.46 ± 0.05	
	Scurve-130-96.json	Ok	1.0	0.47 ± 0.05	
	Scurve-170-96.json	Ok	1.0	0.52 ± 0.04	
	Scurve-210-96.json	Ok	1.0	0.51 ± 0.04	
	Scurve-250-96.json	Ok	1.0	0.58 ± 0.10	
	Scurve-290-96.json	Ok	1.0	0.64 ± 0.13	
	Scurve-330-96.json	Ok	1.0	0.64 ± 0.09	
	Scurve-370-96.json	Ok	1.0	0.49 ± 0.06	

表 7.8: アップロード処理における結果、設定値ファイル添付実行結果と処理時間。図 7.9 より、読み出し試験に対して出力される各ファイルのアップロード実行結果、データ容量と処理時間をまとめた。ここで扱うファイルサイズの合計は 94MB である。図よりファイルのデータ容量が大きいほど処理時間が長いことが分かる。`std_thresholdscan` のようにファイル数が多い項目の場合、合計して大きい処理時間を要することが分かる。全項目において読み出しの設定ファイルにあたる `beforeCfg_chipCfg.json`、`afterCfg_chipCfg.json` のアップロードは、中央データベースの容量制限により失敗していることが分かる。(tex の技術的に caption が複数になってしまふ。)

	Scurve-60-96.json	Ok	0.99	0.51 ± 0.06	
	Scurve-100-96.json	Ok	1.0	0.48 ± 0.05	
	Scurve-140-96.json	Ok	1.0	0.48 ± 0.06	
	Scurve-180-96.json	Ok	1.0	0.52 ± 0.06	
	Scurve-220-96.json	Ok	1.0	0.54 ± 0.05	
	Scurve-260-96.json	Ok	1.0	0.51 ± 0.05	
	Scurve-300-96.json	Ok	1.0	0.66 ± 0.09	
	Scurve-340-96.json	Ok	1.0	0.51 ± 0.06	
	Scurve-380-96.json	Ok	1.0	0.55 ± 0.05	
	sCurve-0.json	Ok	49	1.0 ± 0.1	
	ThresholdDist-0.json	Ok	4.6	0.56 ± 0.06	
	ThresholdMap-0.json	Ok	2,200	4.3 ± 0.1	
	NoiseDist-0.json	Ok	2.3	0.42 ± 0.04	
	Chi2Map-0.json	Ok	2,300	4.5 ± 0.1	
	StatusMap-0.json	Ok	1,300	2.8 ± 0.1	
	StatusDist-0.json	Ok	0.49	0.48 ± 0.04	
	NoiseMap-0.json	Ok	2,200	4.0 ± 0.2	
	Chi2Dist-0.json	Ok	1.1	0.50 ± 0.04	
	TimePerFitDist-0.json	Ok	3.1	0.56 ± 0.12	
	ctrlCfg_ctrlCfg.json	Ok	0.46	0.49 ± 0.05	
	dbCfg_dbCfg.json	Ok	0.60	0.52 ± 0.06	
	siteCfg_siteCfg.json	Ok	0.033	0.54 ± 0.07	
	userCfg_userCfg.json	Ok	0.14	0.60 ± 0.07	
	scanCfg_std_thresholdscan.json	Ok	2.2	0.46 ± 0.03	
	beforeCfg_chipCfg.json	Error	7,200	3.2 ± 0.2	
	afterCfg_chipCfg.json	Error	7,200	3.5 ± 0.1	
std_totscan	MeanTotMap-0.json	Ok	1,900	4.8 ± 0.1	20 ± 0
	SigmaTotMap-0.json	Ok	2,200	4.1 ± 0.2	
	MeanTotDist-0.json	Ok	0.59	0.54 ± 0.07	
	SigmaTotDist-0.json	Ok	1.8	0.47 ± 0.03	
	L1Dist.json	Ok	0.59	0.60 ± 0.08	
	ctrlCfg_ctrlCfg.json	Ok	0.46	0.47 ± 0.03	
	dbCfg_dbCfg.json	Ok	0.60	0.67 ± 0.24	
	siteCfg_siteCfg.json	Ok	0.033	0.59 ± 0.10	
	userCfg_userCfg.json	Ok	0.14	0.56 ± 0.05	
	scanCfg_std_totscan.json	Ok	2.0	0.59 ± 0.10	
	beforeCfg_chipCfg.json	Error	7,200	3.0 ± 0.1	
	afterCfg_chipCfg.json	Error	7,200	3.9 ± 0.3	
std_noisescan	Occupancy.json	Ok	1,300	4.0 ± 0.1	18 ± 0
	NoiseOccupancy.json	Ok	1,300	2.5 ± 0.1	
	NoiseMask.json	Ok	1,300	2.4 ± 0.1	
	ctrlCfg_ctrlCfg.json	Ok	0.46	0.57 ± 0.08	
	dbCfg_dbCfg.json	Ok	0.60	0.55 ± 0.06	
	siteCfg_siteCfg.json	Ok	0.033	0.53 ± 0.04	
	userCfg_userCfg.json	Ok	0.14	0.61 ± 0.07	
	scanCfg_std_noisescan.json	Ok	1.4	0.53 ± 0.05	
	beforeCfg_chipCfg.json	Error	7,200	2.8 ± 0.2	
	afterCfg_chipCfg.json	Error	7,200	3.5 ± 0.1	

表 7.8 より、データサイズの大きいものにアップロード時間がかかっていることがわかる。また添付処理を行うオフセットがあることから、threshold scan のように各容量が大きくなくてもファイル数が多いものにはアップロード時間が合計して多くかかってしまうことがわかる。

これらのことと 4MB 以上の容量を持つファイル添付処理の失敗をなくすために、次のような改善策を考えた。

- 各試験項目に対する結果データ、設定ファイルをそれぞれ Zip ファイルに統合し、圧縮後にアップロードを行う。

こうすることで、アップロードするファイルの容量、数共に削減することができる。圧縮率によってはアップロード処理の失敗もなくすことができると考えた。

これを踏まえアップロードツールを改良し、再び各ファイルの添付処理にかかる時間を測定した。合計処理時間は以下のようにになり、全てのファイルのアップロードに成功した。

$$36 \pm 1[\text{sec}] \quad (7.9)$$

表 7.9: アップロード処理改善後における結果、設定値ファイル添付実行結果と処理時間。各試験結果毎に結果ファイル、設定値ファイルを Zip ファイルにまとめアップロードする処理とした。これにより、ファイル数、容量の削減に成功し、アップロード時間が改善した。全てのファイルのアップロードに成功していることが分かる。

読み出し項目	ファイル名	実行結果	容量 [KB]	処理時間 [sec]	全体 [sec]
std_digitalscan	std_digitalscan_datafiles.zip	Ok	10	0.77 ± 0.18	1.9 ± 0.2
	std_digitalscan_configfiles.zip	Ok	56	1.1 ± 0.1	
std_analogscan	std_analogscan_datafiles.zip	Ok	46	1.0 ± 0.2	2.2 ± 0.3
	std_analogscan_configfiles.zip	Ok	58	1.2 ± 0.2	
std_thresholdscan	std_thresholdscan_datafiles.zip	Ok	1,500	2.6 ± 0.1	3.5 ± 0.1
	std_thresholdscan_configfiles.zip	Ok	190	0.86 ± 0.08	
std_totscan	std_totscan_datafiles.zip	Ok	730	1.7 ± 0.2	2.5 ± 0.2
	std_totscan_configfiles.zip	Ok	190	0.83 ± 0.15	
std_noisescan	std_noisescan_datafiles.zip	Ok	19	0.56 ± 0.07	1.7 ± 0.1
	std_noisescan_configfiles.zip	Ok	190	1.1 ± 0.1	

生産時における見積もり

上記の見積もりと同様に、改善後のツールにおけるアップロード時間の見積もりを行った。表 7.9において、添付処理に対応する処理 4、5 以外は同じとする。改良後の処理 4、5 の処理時間は、

$$\begin{aligned} \text{FE チップ処理} &: \left\{ (2.2 + 6.7 + 5.1) \pm \sqrt{(0.1)^2 + (1.1)^2 + (0.3)^2} \right\} \times 4 \\ &= 56 \pm 5[\text{sec}] \end{aligned} \quad (7.10)$$

$$\begin{aligned} \text{合計} &: (56 \pm 5)[\text{sec}] + (14 \pm 0) + (0.88 \pm 0.14) + (1.8 \pm 0.1) \\ &= 1.2 \pm 0.1[\text{min}] \end{aligned} \quad (7.12)$$

約 1 分でアップロードを完了できる見積もりとなった。改善前に比べて 15% の処理時間となった。現在は改善後の方針を用いたツールを提供している。

1174 ファイル添付以外の処理

1175 ここではファイル添付処理に関する処理時間検討を行なったが、以下にあげたそれ以外の処理について言及する。

- 1177 1. ローカルデータベース内部処理.
1178 2. 中央データベースへの接続、認証処理.
1179 3. ファイル添付以外の処理に関する、中央データベース API 使用.
1180 4. ネットワーク速度の改善.

1181 項目1に関して、読み出し試験におけるローカルデータベースの内部構造は世界的に使われているものである。内部構造の変更をすることなく改善を図りたいという開発方針から、この項目に関する処理時間削減の検討を行わなかった。また図7.9よりローカルデータベース内部処理は十分に小さいものであったため、検討の必要がないと考えた。

1185 項目2に関して、中央データベースへの接続、認証処理は中央データベースのAPI及び中央データベースの内部構造によるものである。この部分の処理を改善するには、これらの変更を検討する必要がある。

1187 項目3に関して、アップロード処理では以下の処理の際に中央データベースのAPIを使用している。

- 1188 ● モジュール情報の取得.
1189 ● 結果ページの作成(モジュールとFEチップで5回分)

1190 これらの処理は必要であるため、APIの使用回数の減らし処理時間を削減することは難しい。

1191 項目4に関して、中央データベースと通信する全ての処理はのネットワーク速度に依存していると考えられる(付録E)。KEKのサーバーが置かれているネットワーク環境の改善により、本ツールの処理速度も改善すると考えられる。

1194 第8章

1195 まとめ

1196 8.1 本論文のまとめ

1197 HL-LHCに向けてATLAS内部飛跡検出器の総入れ替えを予定しており、これに向けてピクセルモ
 1198 デュールを世界で10,000台生産する予定である。各モジュールに対して品質試験を行い、全てのモジュール
 1199 及び品質試験の結果は中央データベースに保存する。

1200 本研究では、この生産及び品質試験に向けてデータベースシステムの構築を行った。各生産現場にて
 1201 データ保存、管理をするローカルデータベースを確立し、品質試験結果検索や中央データベースとの同期
 1202 機能など、生産時に必要となる諸ツールの開発を行った。

1203 開発した諸ツールを含め、生産において必要な機能の確認を行った。本番を想定したソフトウェア、
 1204 ハードウェアのセットアップと各ツールの処理実行を達成し、機能が使用可能であることを確認した。

1205 主な開発項目の1つ目として品質試験検索機能を述べた。開発当初はデータベース内部構造により、
 1206 試験結果数 n に対して処理時間が $O(n^2)$ かかってしまう問題が発生した。MongoDB 内に新しいコレク
 1207 ションを設け検索に必要な情報を予め1つの場所に保持しておくことにより、処理時間の改善に成功し
 1208 た。実際に処理時間の測定を行い、データ数の増加に対しても検索機能が不都合なく使えることを確認し
 1209 た。本番を想定した見積もりを行い、84,000件のデータ数に対して $2.6 \pm 0.1[\text{sec}]$ で処理が実行できる見
 1210 込みであり、生産時において十分に使える機能であることを確認した。

1211 2つ目に中央データベースとの同期ツールを開発した。世界的に使われるツールであり、全ての生産現
 1212 場でこのツールをサポートするために中央データベースへの通信処理時間調査を KEK、LBL、CERN
 1213 のサーバーを用いて行った。KEKのサーバーを用いた場合に最も時間がかかるることを確認し、このサー
 1214 バーにおいて十分に使うことができる機能開発を達成すれば世界的に問題がないと考えた。開発した中央
 1215 データベース同期ツールについて KEK サーバーを用いて処理速度測定を行った。モジュール情報のダウ
 1216 ンロード機能に関して、モジュール1つあたり $4.0 \pm 0.4[\text{sec}]$ の処理時間がかかるることを確認した。処理
 1217 の詳細を調査すると、モジュールや構成部品の情報取得するために行っている中央データベース API の
 1218 使用に多くの時間がかかっていた。処理時間の改善策をいくつか考案し、それぞれについて見積もりを行っ
 1219 た。読み出し試験結果のアップロード機能に関して、開発当初はモジュール1つに対して結果アップロー
 1220 ド処理時間の見積もり値が $7.9 \pm 0.1[\text{min}]$ であった。処理時間改善に向けて処理の詳細を調査し、結果
 1221 ファイルの添付処理に大きく時間が要していることが分かった。ファイル添付についての詳細を調べる
 1222 と、添付処理時間がファイル数とファイル容量に依存していた。このことから結果ファイルを ZIP ファ
 1223 イルにまとめ、圧縮しアップロードを行うことで処理時間の改善を図った。ここで圧縮前後のファイル容
 1224 量は、それぞれ 94、3.9[MB/1FE チップ] となった。結果として処理時間の改善に成功し、その見積もり

1225 値が $1.2 \pm 0.1[\text{min}]$ となった。またファイル添付以外の処理に関しても削減の余地がないかを検討した。

1226 8.2 現状と今後の課題

1227 8.2.1 ソフトウェアリリースとユーザサポート

1228 本論文で述べたツールの他に、読み出し試験コマンド統括ソフト、品質試験結果アップロード用ソフト
1229 などの開発もチームとして行っている。全てのソフトウェアを含めて、品質試験のデータ管理を達成する
1230 ようなアプリケーションスイートを目指している。2020年12月9日にファーストバージョンのリリース
1231 を行い、いくつかの機関で全体のシステム及びソフトウェアが使われている現状である。

1232 またCERNで行ったチュートリアルを経て、世界的に機能普及が進んでいる。そのためユーザサポート
1233 としてソフトウェア使用のためのドキュメント[7-1]の作成、整備も行っている。開発者の連絡先や
1234 ローカルデータベース専用掲示板へのリンクもドキュメントに記している。何か問題が生じた時などに簡単
1235 に問い合わせができる仕組みを整えている。

1236 8.2.2 開発課題

1237 本研究では検索機能や同期機能など、読み出し試験を対象とした機能を重点的に開発した。ローカル
1238 データベース開発は、読み出し試験の結果を管理したいという要求から始まり、現在はそれ以外の品質試
1239 験も含め、全ての結果や組み立て工程の管理も目標としている。今後の開発課題として以下の機能をあ
1240 げる。

- 1241 • 読み出し試験以外(外観検査、平坦性測定等)の結果同期機能.
- 1242 • 中央データベースからローカルデータベースへ品質試験結果の同期.
- 1243 • 品質試験結果解析とモジュール選別機能.
- 1244 • 組み立て工程管理を世界的にサポート.

1245 最後の項目に関して、モジュールの組み立て工程は各機関ごとに異なるため、全ての現場における工程
1246 を調査しそれをサポートするシステムを実装する必要がある。例えば、日本では「ペアモジュール・フレ
1247 キシブル基板貼り付け」工程の後に、モジュール冷却のための構造である「セル搭載」を行う予定であり、
1248 これは他の組み立て機関とは異なる。各地域における柔軟な生産を許しているため、組み立て工程は世界
1249 的に統一されていない。データベースシステムでは多様な組み立て工程に対応できるシステムとする必要
1250 がある。

1251 このシステム実装において、中央データベースには選択可能な組み立て工程を定義する機能が存在し、
1252 これを使用することを考えている。そのイメージを図8.1に示す。

1253 以下の開発項目をあげる。

- 1254 • 選択式工程の機能を使い、全ての組み立て機関における工程を中央データベース上に定義.
- 1255 • 本研究で開発した同期ツールを拡張、ローカルデータベース上にも中央データベースと同様の組み
1256 立て工程構造を保持.

1257 これらを達成することにより、全ての機関における組み立て工程情報の管理ができるようになると考え
1258 ている。



図 8.1 中央データベースにおける選択可能な組み立て工程のイメージ。図に示すように中央データベースでは選択可能な組み立て工程を定義することができる。図ではセル搭載が選択可能となっていて、ペア・基板貼り付け工程の後に、どちらの工程に進むのかを選択できる。この機能を用いて世界的に多様な組み立て工程をサポートすることを考えている。

1259 付録 A

1260 シリコン検出器の原理

1261 A.1 半導体 [8]

1262 固体は、絶縁体、半導体、導体の 3 つに大別できる。物質の電気伝導度に関して、絶縁体は非常に低い
 1263 値 ($10^{-18} \sim 10^{-8}$ S/cm)、導体は高い値 ($10^4 \sim 10^6$ S/cm) を持つ。半導体の電気伝導度はこれらの中間
 1264 であり、温度、光、磁界および微量の不純物に対し非常に敏感である。この特徴のために半導体はエレク
 1265 トロニクスにおける最も重要な材料の 1 つになっている。半導体は元素半導体と化合物半導体に分けら
 1266 れ、多くの物質がその候補となる。元素半導体の中で代表的なものとして Si があげられ、ATLAS ピク
 1267 セル検出器に使われる半導体は Si がベースとなっている。不純物が入っていない、全ての原子が Si の半
 1268 導体を真性半導体と呼ぶ。真性半導体中の Si は 4 つの Si と共有結合を構成し、結晶を作る。(図 A.1)
 1269 真性半導体に対し、As などの最外殻電子を 5 つもつ原子を不純物としてドープしたものを n 型半導体、
 1270 B などの 3 つのものをドープしたものを p 型半導体と呼ぶ。それぞれキャリアとして電子、ホールを持
 1271 つことになり、キャリア移動の特性を組み合わせて様々なデバイスに応用することができる。

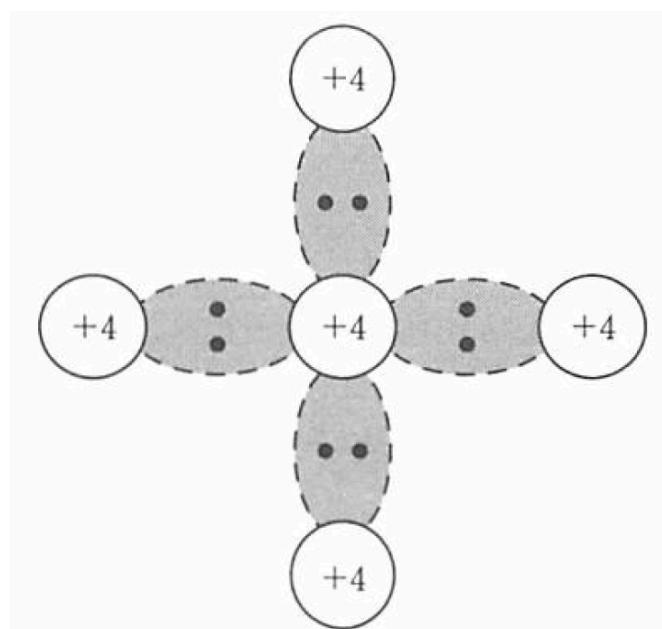


図 A.1 真性半導体中のシリコン [8]

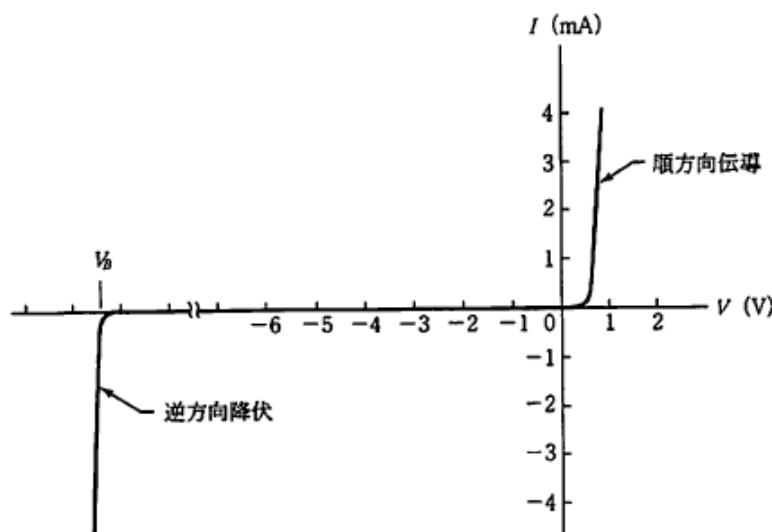


図 A.2 pn 接合の電流 - 電圧特性 [8]



図 A.3 空乏層 [8]

A.1.1 pn 接合

n 型半導体と p 型半導体を接合し、その接合部を pn 接合と呼ぶ。この接合は各種半導体素子で様々な形で応用されており、ピクセル検出器にも用いられている。

pn 接合の最も重要な特徴は特定の方向にだけ電流が流れやすい整流性である。図 A.2 に示すように正電圧をかけると電流は急速に増加する。逆方向にかけた場合、始めのうちは電流はほとんど流れない。ある臨界電圧に達すると電流は急激に増大する。

逆方向電圧をかけた場合、図 A.3 に示すように pn 接合付近はキャリアが存在しない空乏層領域が形成される。この時、それぞれの半導体のエネルギー準位に差が生じている状態となっている。印加電圧 V と空乏層幅 W は以下のような関係がある。

$$W \propto \sqrt{V} \quad (\text{A.1})$$

1281 A.2 検出原理

1282 荷電粒子が物質中を通過するとき、以下の Bethe-Bloch の公式によってエネルギーを損失する [9]。

$$-\left\langle \frac{dE}{dx} \right\rangle = K z^2 \frac{Z}{A} \frac{1}{\beta^2} \left(\frac{1}{2} \ln \frac{2m_e c^2 \beta^2 \gamma^2 T_{max}}{I^2} - \beta^2 + \dots \right) \quad (A.2)$$

$\frac{dE}{dx}$: 荷電粒子のエネルギー損失量 [eV · g⁻¹ · cm²]
 K : $4\pi N_A r_e^2 m_e c^2 = 0.307075$ [MeVcm²]
 z : 荷電粒子の電荷量
 Z : 物質の原子番号 (Si14)
 A : 物質の原子量 (Si28)
 $m_e c^2$: 電子の静止エネルギー (0.511MeV)
 β : 光速を 1 とした入射粒子の速度
 γ : ローレンツ因子 $1/\sqrt{1 - \beta^2}$
 I : 励起エネルギーの期待値 (シリコン 137eV)

(A.3)

1283 また T_{max} は質量 M の入射粒子による 1 つの電子への最大運動エネルギー移行であり、以下の式で書
1284 ける。

$$T_{max} = \frac{2m_e c^2 \beta^2 \gamma^2}{1 + 2\gamma m_e / M + (m_e / M)^2} \quad (A.4)$$

1285 荷電粒子が半導体を通過したとき、そのエネルギー損失量に応じて電子・ホール対が生成し、その量を
1286 測定することができる。

1287 例として、大きい速度 ($\beta\gamma \gg 1$) を持つ荷電粒子が、300μm の厚さを持つシリコン半導体を通過した
1288 時の生成キャリア数及び信号強度を計算する。

1289 荷電粒子の速度が十分に早い場合、式 A.2 は物質の密度 ρ を用いて以下のように近似できる [34]。

$$-\left\langle \frac{dE}{dx} \right\rangle = 1 \sim 2\rho [\text{MeV}/\text{cm}] \quad (A.5)$$

1290 シリコンの密度を 2.33 g/cm³、シリコンにおける 1 キャリア対生成に必要な平均電離エネルギー
1291 3.62 eV より、300μm の厚さを持つシリコン半導体を通過した時の生成キャリア数は以下のように計算
1292 できる。

$$1 \sim 2 \times 2.33 \times (3 \times 10^4) \times 1/3.62 \sim 20,000 \sim 39,000 \quad (A.6)$$

1293 素電荷 $e = 1.6 \times 10^{-19}$ C より、信号強度は以下のようになる。

$$20,000 \sim 39,000 \times 1.6 \times 10^{-19} \sim (3.2 \sim 5.1) \times 10^{-15} [\text{C}] \quad (A.7)$$

付録 B
1294

1295 RD53A の回路図とフレキシブル基板

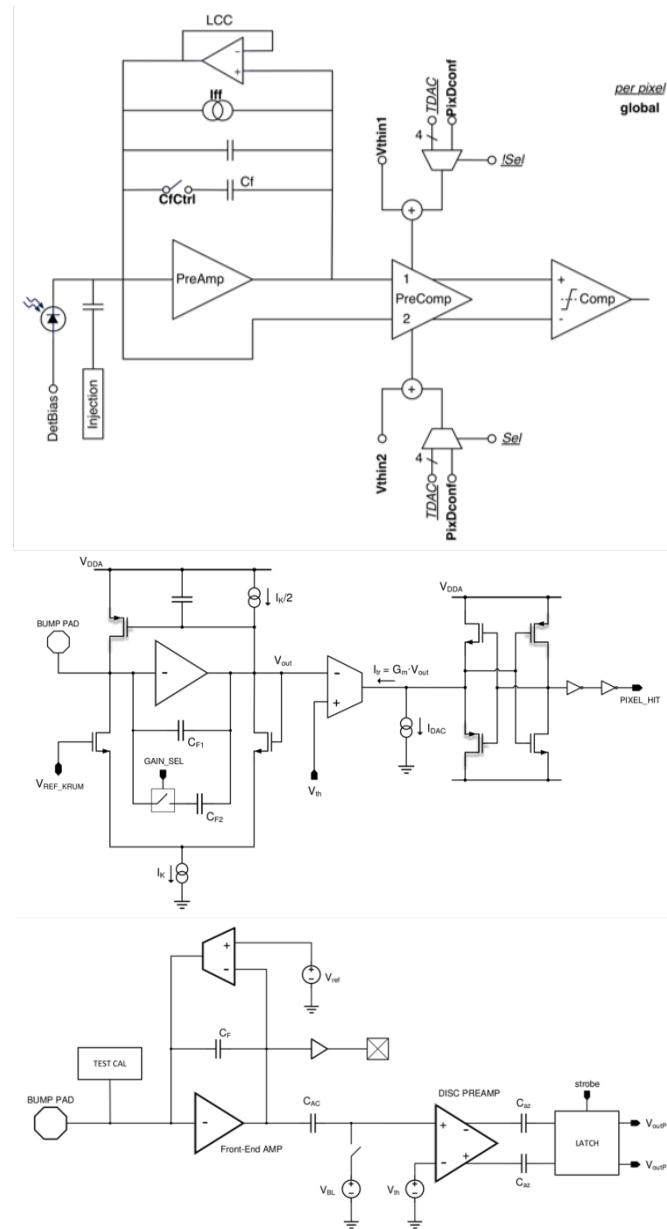


図 B.1 アナログフロントエンド [8]

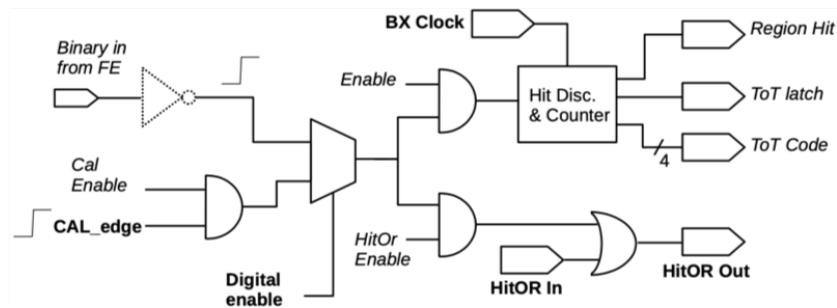


図 B.2 デジタルフロントエンド [8]

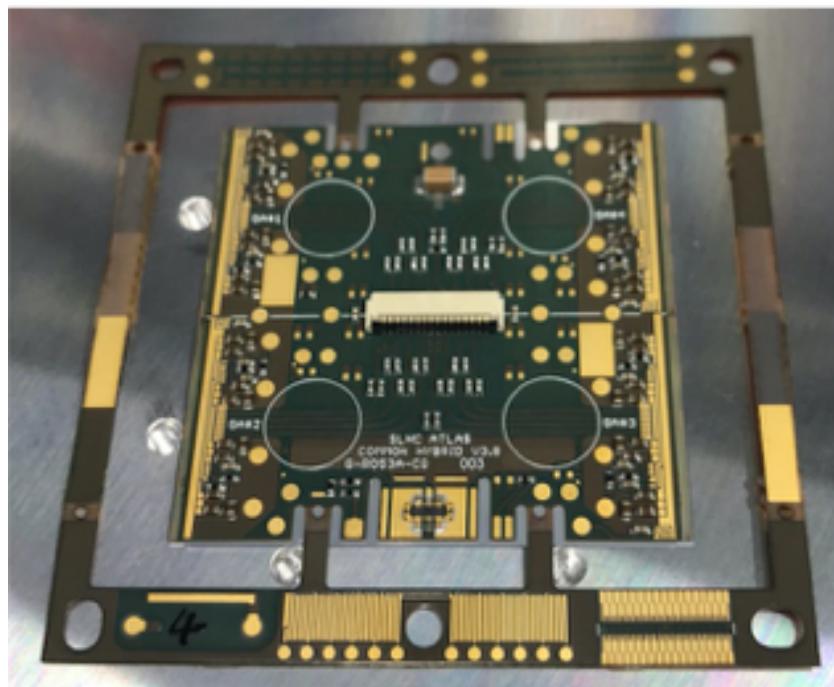


図 B.3 フレキシブル基板

付録 C

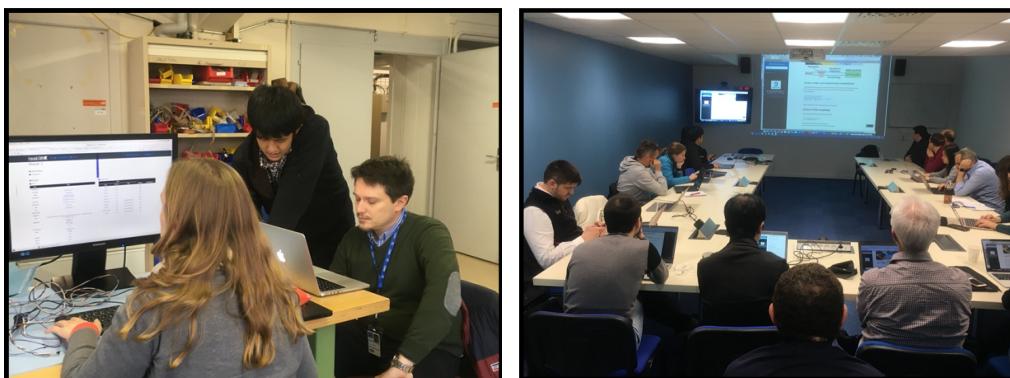
ローカルデータベースのチュートリアル と普及状況

ローカルデータベースの機能の普及を目的として、2020年2月にCERN研究所にてシステムのチュートリアルを行った。このチュートリアルは以下のような2つのセッションに分けて行った。

- 参加者が実際にサーバーの設定、各ソフトウェアのインストールを行いながら機能を実践するセッション（2月3日から6日まで）
- 私が参加者の前で実際に機能を実践し、システムや使い方に対して議論を行うセッション（2月7日）

それぞれのセッションの様子を図C.1に示す。数多くの議論を行い、有益なフィードバックを得ることができた。また品質試験の流れにおいて、一連の機能確認をすることができた。

これを経て現在ローカルデータベースは世界18箇所にて導入され、試験運用が開始している。将来的には全組み立て機関で使うことが決定しており、それに向けたシステム開発、サポートが必要となっている状況である。ローカルデータベースについて、導入及び試験運用を行っている機関を以下に示す。また世界地図をC.2に示す。



図C.1 ローカルデータベースシステムチュートリアルの様子。2020年2月にCERNでローカルデータベースシステムのチュートリアルを行った。参加者が実際にシステムの設置、機能実行を行うハンズオンセッション（左図）と、参加者の前で実際に機能の動作をみせ、議論を行うハンズオフセッション（右図）に分けて行った。システムに有益な情報を獲得したと共に、システムの機能普及に成功した。

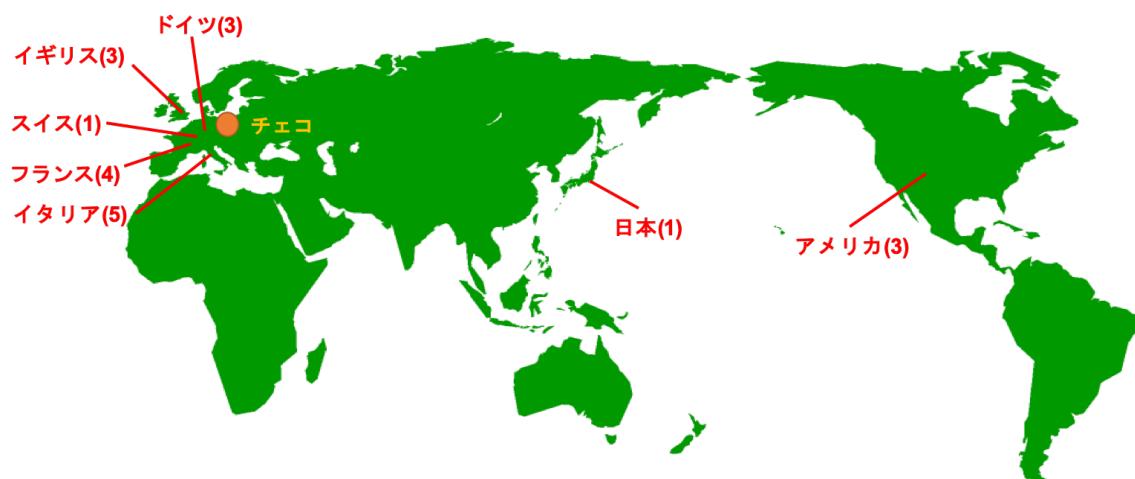


図 C.2 ローカルデータベースシステム導入及び試運転場所。赤文字は設置している地域、括弧内の数字はその地域におけるシステム導入場所の数を示している。2020 年 11 月現在、ローカルデータベースシステムは世界 18 の機関で試験運転がなされている。日本を除いてその多くはヨーロッパとアメリカに位置していることが分かる。

- 1311 ● 高エネルギー加速器研究機構 (KEK), 日本
- 1312 ● 欧州原子核研究機構 (CERN), スイス
- 1313 ● University of Liverpool, イギリス
- 1314 ● University of Oxford, イギリス
- 1315 ● University of Glasgow, イギリス
- 1316 ● Paris-Saclay University, フランス
- 1317 ● パリ第 6 大学, フランス
- 1318 ● フランス国立科学研究中心, フランス
- 1319 ● University of Grenoble, フランス
- 1320 ● University of Gottingen, ドイツ
- 1321 ● University of Siegen, ドイツ
- 1322 ● University of Genoa, イタリア
- 1323 ● University of Salento, イタリア
- 1324 ● University of Milan, イタリア
- 1325 ● University of Udine, イタリア
- 1326 ● University of Trento, イタリア
- 1327 ● University of Oklahoma, アメリカ
- 1328 ● Argonne National Laboratory, アメリカ
- 1329 ● Lawrence Berkeley National Laboratory(LBL), アメリカ

1330 付録 D

1331 モジュール生産状況の解析

1332 上述したデータベースシステムを使って、モジュール生産状況の解析を行うことができる。モジュール
 1333 の組み立て工程は各生産場所のローカルデータベース上に記録され、組み立て工程ごとに中央データベー
 1334 スへ同期される。そのため現在組み立てが行われている全てのモジュールの現在工程を中央データベース
 1335 上で取得できることができ、この情報を用いて世界的な生産状況の解析を行うことができる。現在は生産
 1336 は行われていないが、想定している解析結果のイメージを図 D.1 に示す。

1337 生産数や生産レートのモニタリングを行うことで、今後の生産計画や問題解決に役立てることが可能で
 1338 ある。

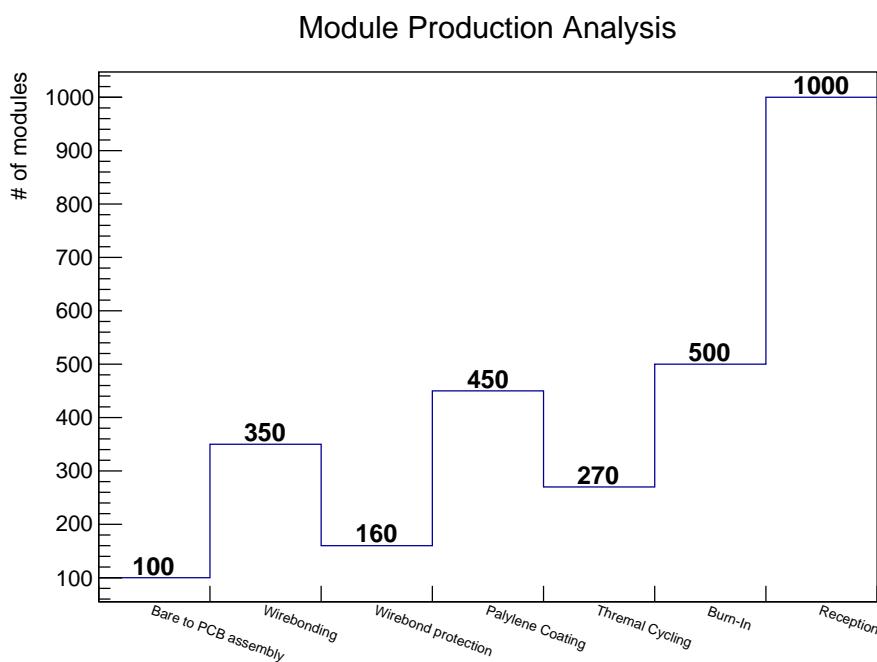


図 D.1 生産時のモジュール組み立て状況解析の例。ローカルデータベースにて組み立て工程は管理
 され、工程毎に中央データベースに同期されるため、生産時には全てのモジュールの現工程を中央データ
 ベースで取得できる。図の例のように各段階におけるモジュール数を見ることで生産状況の確認がで
 きる。さらにある期間ごとに工程を取得することで生産レートなども計算することができ、今後の生
 産計画やモジュール選別の参考とすることができる。

付録 E

ファイル送信時におけるデータ容量と処理時間の関係について

1342 処理速度遅延の原因として以下をあげる。

- 1343 ● サーバーの読み書き速度.
- 1344 ● サーバーのファイル転送アルゴリズム.
- 1345 ● サーバーのファイル転送時におけるパケットサイズ.
- 1346 ● サーバー間のネットワーク上の距離.
- 1347 ● サーバー間ネットワークの処理性能.

1348 KEK と LBL に設置されているサーバーにおいて、中央データベースへのファイル送信処理時間に差
1349 が出る理由について考察する。

1350 初めに、ファイル送信時におけるデータ容量と処理時間の関係は、線形性を示さない。図 E.1 は KEK
1351 から LBL のサーバーに scp コマンドを用いてファイル送信を行い、データ容量と処理時間の関係を取得
1352 したものである。赤線が線形フィットであるが、測定点は優位にずれている。これは TCP 通信において
1353 パケットの送信に輻輳制御 [36] と呼ばれる技術が使われており、データ送信量を変化させながら情報通信
1354 を行っているためである。

1355 scp によるファイル送信を以下の 2 つの場合に対して行い、ファイル容量と処理時間の関係を取得した。

- 1356 1. KEK から LBL
- 1357 2. LBL から KEK

1358 結果を図 E.2 に示す。ここで処理時間に差が生まれる原因について調査したものを以下に示す。

- 1359 ● 読み書き速度を測定したところ同程度であった。
- 1360 ● 輻輳制御アルゴリズムは Cubic をどちらも使用していた。
- 1361 ● パケットサイズは変わらなかった。
- 1362 ● ping(111msec) は変わらなかった。

1363 よってサーバ間ネットワークに差異があると考えられる。一般的には上りより下りの方が太いと考え
1364 と、KEK ローカルの上りネットワークの性能が、LBL ローカルの上りネットワークと比べて悪いと考え
1365 られる。これにより、ファイル送信時間に差が生まれている。

1366 次に、下の 2 つの場合に対して行い、ファイル容量と処理時間の関係を取得した。

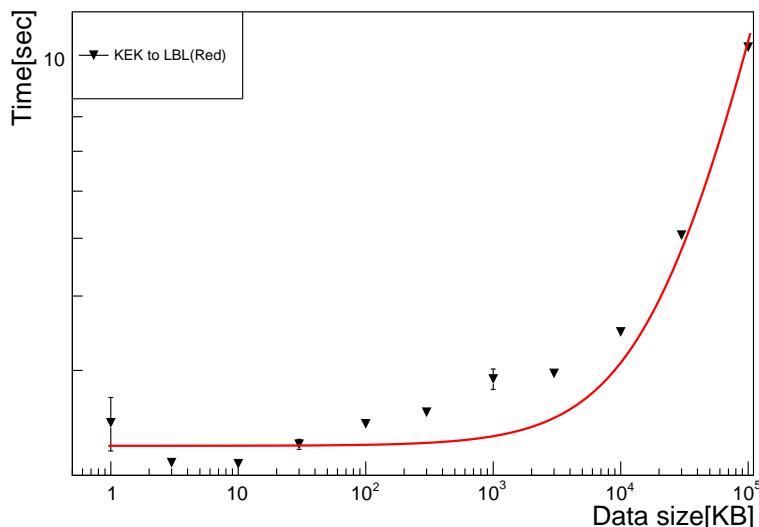


図 E.1 添付するファイルサイズと処理時間の関係

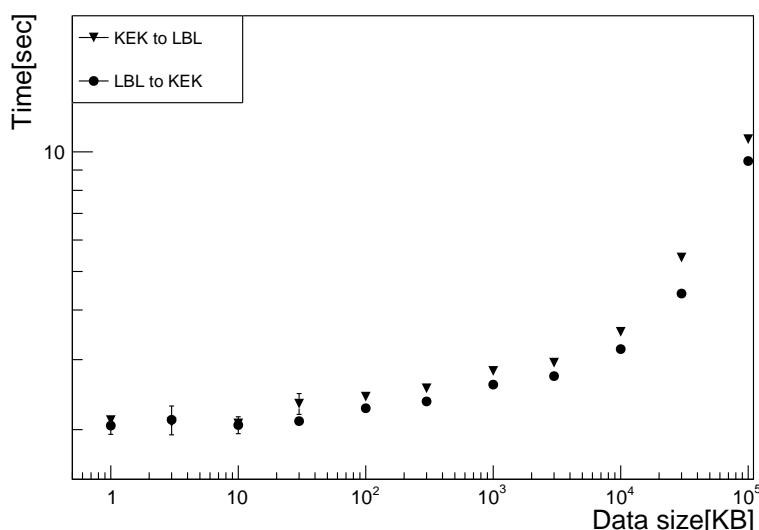


図 E.2 KEK、LBL 間のファイル送信

- 1367 1. KEK から CERN(Lxplus)
1368 2. LBL から CERN(Lxplus)

1369 KEK ローカルの上りネットワークは細いため、処理時間に差が生まれる。加えてこの場合はサーバ間
1370 の距離による遅延も含まれていると考えられる。ping による反応時間が測定 1 では 170msec 程度なのに
1371 対し、測定 2 は 150msec 程度であった。そのためこれも処理速度遅延に影響していると考えられる。
1372 CERN と中央データベースが地理的に近い距離であることを考慮し、上述したことをまとめると KEK
1373 と LBL の間で処理時間の差が生まれる要因は以下であると考えた。

- 1374 ● ローカルネットワークの性能。
1375 ● ネットワーク上の距離差。

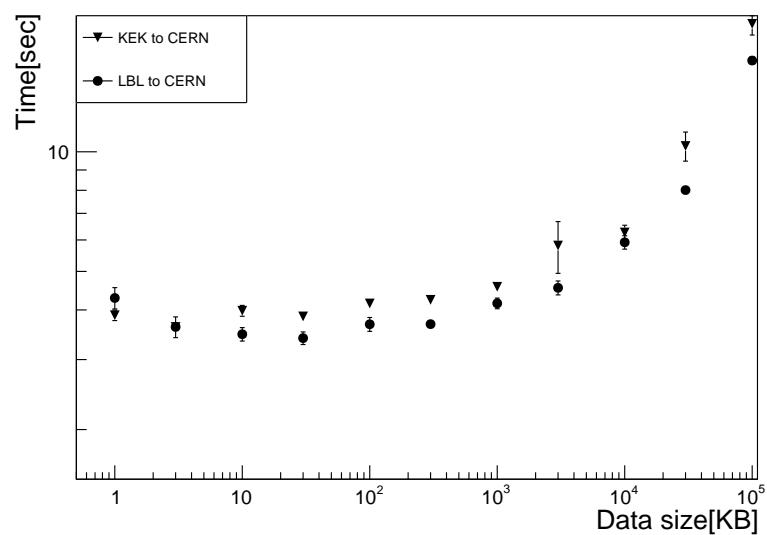


図 E.3 KEK、LBL と CERN 間のファイル送信

参考文献

- 1376 [1] Damerau,H et al. “LHC Injectors Upgrade Technical Design Report”. CERN Document server.
 1377 2016-05
 1378 [\(2020-12\)](https://cds.cern.ch/record/2153863)
- 1379 [2] Georges Aad et al. “The ATLAS Experiment at the CERN Large Hadron Collider”. Semantic
 1380 Scholar. 2008-8
 1381 <https://www.semanticscholar.org/paper/The-ATLAS-Experiment-at-the-CERN-Large-Hadron-Aad-Gr...>
 1382 7d771b20731969fe10c267465582ee60e9383db3,(2020-12)
- 1383 [3] ATLAS Collaboration. “The upgraded Pixel Detector of the ATLAS Experiment for Run 2 at
 1384 the Large Hadron Collider”. ScienceDirect. 2016-9
 1385 [\(2020-12\)](https://doi.org/10.1016/j.nima.2016.05.018)
- 1386 [4] Apollinari, G;Bjar Alonso, I; Brning, O; Lamont, M; Rossi, L. “High-Luminosity Large Hadron
 1387 Collider (HL-LHC) : Preliminary Design Report”. CERN Document Server. 2015-12
 1388 [\(2020-12\)](https://cds.cern.ch/record/2116337)
- 1389 [5] The HL-LHC project. “The HL-LHC project”. CERN Accelerating science. 2020-8
 1390 <https://hilumilhc.web.cern.ch/content/hl-lhc-project>,(2020-12)
- 1391 [6] ATLAS Collaboration. “Technical Design Report for the ATLAS Inner Tracker Pixel Detector”.
 1392 CERN Document Server. 2018-8
 1393 [\(2020-12\)](https://cds.cern.ch/record/2285585)
- 1394 [7] ATLAS Collaboration. “Observation and measurement of Higgs boson decays to WW with the
 1395 ATLAS detector”. CERN Document Server. 2020-6 更新.
 1396 [\(2020-1\)](https://cds.cern.ch/record/1975394)
- 1397 [8] “Semiconductor Devices Physics and Technology”.
 1398 S.M. Sze 著, 南日康夫・川辺光央・長谷川文夫訳, 産業図書, 2015 年 3 月第 2 版第 11 刷発行.
- 1399 [9] “Pixel Detectors”.
 1400 Rossi, L., Fischer, P., Rohe, T., Wermes, N. Springer, 2006-7-8
- 1401 [10] “トータルドーズ効果”. 天文学辞典. 2018-3
 1402 <https://astro-dic.jp/total-dose-effect/>,(2020-12)
- 1403 [11] Danilo Giugni. “General approach for thermal-mechanics QA and QC”. CERN Indico. 2019-12
 1404 <https://indico.cern.ch/event/860761/contributions/3661710/>,(2020-12)
- 1405 [12] Lakmin Wickremasinghe. “Pixel modules and Hybridization:First results from RD53A produc-
 1406 tion”. CERN Indico. 2020-9
 1407 <https://indico.cern.ch/event/950039/contributions/4024890/attachments/2108087/>

- 1409 3546372/QuadModuleTestResults-ITkWeek.pdf,(2020-12)
- 1410 [13] Timon Heim. "YARR - A PCIe based Readout Concept for Current and Future ATLAS Pixel
1411 Modules". IOP Science. 2017
<https://iopscience.iop.org/article/10.1088/1742-6596/898/3/032053>,(2020-12)
- 1412 [14] "物理数学 II".
1413 西森秀穂 著, 丸善出版, 平成 27 年 9 月 30 日発行.
- 1414 [15] Meng, Lingxin. "RD53A Module Testing Document". CERN Document server. 2020-9
<https://cds.cern.ch/record/2702738>,(2020-12)
- 1415 [16] "MongoDB: The most popular database for modern apps". MongoDB, Inc.
<https://www.mongodb.com/2>,(2020-12)
- 1416 [17] "Welcome to Flask". Flask Documentation.
<https://flask.palletsprojects.com/en/1.1.x/>,(2020-12)
- 1417 [18] 窪田ありさ. "HL-LHC ATLAS 実験用新型ピクセル検出器の系統評価と量産時に向けた試験管理シ
1418 ステムの開発" CERN Box. 2020-3
<https://cernbox.cern.ch/index.php/s/BdhvSTAuuE5xHxt>,(2020-1)
- 1419 [19] "Databases and Collections". MongoDB Manual.
<https://docs.mongodb.com/manual/core/databases-and-collections/>,(2020-12)
- 1420 [20] "PyMongo 3.11.2 Documentation". PyMongo 3.11.2 Documentation.
<https://pymongo.readthedocs.io/en/stable/>,(2020-12)
- 1421 [21] "ROOT: analyzing petabytes of data, scientifically.". ROOT Team.
<https://root.cern>,(2020-12)
- 1422 [22] "QC Software doc". ITk Docs.
https://itk.docs.cern.ch/pixels/qc__software/rd53a__demo__flow/,(2020-1)
- 1423 [23] "InfluxDB: Purpose-Built Open Source Time Series Database". influxdata.
<https://www.influxdata.com>,(2020-12)
- 1424 [24] "Grafana: The open observability platform". Grafana Labs.
<https://grafana.com>,(2020-12)
- 1425 [25] "PySerialComm". GitLab.
<https://gitlab.cern.ch/solans/PySerialComm>,(2020-12)
- 1426 [26] "RD53A Single Chip Card Configuration". CERN twiki. 2018-5-14
https://twiki.cern.ch/twiki/pub/RD53/RD53ATesting/RD53A__SCC__Configuration.pdf,(2020-12)
- 1427 [27] Renesas Electronics Corporation. "GPIO". RENESAS.
<https://www.renesas.com/jp/ja/support/engineer-school/mcu-programming-peripherals-01-gpio>,(2020-12)
- 1428 [28] Microchip Technology. "2.7V Dual Channel 10-Bit A/D Converter with SPI/TM Serial Interface".
秋月電子通商. 2006-08
<https://akizukidenshi.com/download/ds/microchip/mcp3002.pdf>,(2020-12)
- 1429 [29] "Raspberry Pi 3 Model B+". RASPBERRY PI FOUNDATION.
<https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/>,(2020-12)
- 1430 [30] "E3640A E3649A Programmable DC Power Supplies - Data Sheet". Keysight Technologies.

- 1450 2018-3
1451 [\(2020-12\)](https://www.keysight.com/jp/ja/assets/7018-06827/data-sheets/5968-7355.pdf)
1452 [31] “XpressK7-160-Gen2”. Mouser Electronics.
1453 [\(2020-12-22\)](https://www.mouser.jp/ProductDetail/ReFLEX-CES/XpressK7-160-Gen2?qs=rrS6PyfT74eSJLUPLu1P5g%3D%3D)
1454 [32] Wielers, Monika. “Pixel production database serial numbering scheme”. CERN Document
1455 Server. 2020-12-07
1456 [\(2020-12\)](https://cds.cern.ch/record/2728364)
1457 [33] “LocalDB docs”. LocalDB docs.
1458 [\(2020-12\)](https://localdb-docs.readthedocs.io/en/top/)
1459 [34] “Selected Exercises in Particle and Nuclear Physics”.
1460 Lorenzo Bianchini, Springer, 2017-9 published.
1461 [35] Garcia-Sciveres, Maurice. “The RD53A Integrated Circuit”. CERN Document Server. 2017-10
1462 [\(2020-12\)](https://cds.cern.ch/record/2287593)
1463 [36] “TCP 技術入門-進化を続ける基本プロトコル”.
1464 安永遼真, 中山悠, 丸田一輝著, WEB+DB PRESS plus, 2019 年 7 月 6 日発売.
1465

謝辞

1467

図目次

1468

1469	1.1 加速器の全体像	2
1470	1.2 極角 θ と擬ラピディティ η の関係図	2
1471	1.3 ATLAS 検出器の全体像	3
1472	1.4 内部飛跡検出器の全体像	4
1473	1.5 ピクセル検出器の全体像	4
1474	1.6 ピクセルモジュールの全体像。	5
1475	1.7 HL-LHC 運転計画	6
1476	1.8 ITk の全体像	7
1477	1.9 ITk の断面図	7
1478	1.10 VBF イベントの図	9
1479	2.1 ピクセルモジュールの構成	10
1480	2.2 新型ピクセルモジュールに搭載するシリコンセンサーの断面図	11
1481	2.3 ToT の概念図	11
1482	2.4 RD53A	12
1483	2.5 ピクセルモジュールにおける信号伝達の様子	13
1484	2.6 搭載するモジュールのプロトタイプと ITk における配置。	14
1485	3.1 組み立て工程のイメージ図	16
1486	3.2 外観検査の様子	17
1487	3.3 平坦性測定の様子。	17
1488	3.4 センサー電流-電圧特性結果の例	18
1489	3.5 FE チップ電流-電圧特性試験結果の例。	18
1490	3.6 読み出し試験の様子	19
1491	3.7 デジタル回路読み出しにおける <i>Occupancy</i> 分布の例。	22
1492	3.8 入射電荷量と <i>Occupancy</i> の関係	22
1493	3.9 Threshold 値とノイズの分布。	23
1494	3.10 ノイズ占有率測定における <i>NoiseOccupancy</i> 分布の例。	24
1495	3.11 組み立て工程と対応する品質試験一覧	26
1496	4.1 ローカルデータベースシステムの概要	29
1497	4.2 MongoDB の構造の例 [19]	31
1498	4.3 先行研究により設計された読み出し試験結果の MongoDB 内部構造	32

1499	4.4	ウェブアプリケーション処理のイメージ	33
1500	4.5	品質試験結果ページの例	34
1501	4.6	中央データベースにおけるモジュールの種類と構造	35
1502	4.7	中央データベース内におけるモジュール構造の一例 (Quad モジュール)	36
1503	4.8	同期ツールのデータ通信のイメージ	37
1504	4.9	同期機能の概要	37
1505	4.10	ウェブアプリケーションにおけるコメント機能	39
1506	4.11	ウェブアプリケーションにおけるタグ機能	39
1507	4.12	結果選択画面及び組み立て工程表示の例	43
1508	4.13	ピクセル解析ツールにおけるファイル統合処理のイメージ	44
1509	4.14	Tree ファイルとそのデータ保持	45
1510	4.15	ピクセル解析ツールの処理の流れ	45
1511	4.16	ウェブアプリケーションにおける検索機能の様子	46
1512	4.17	各モジュールにおけるデータベースシステム操作の流れ	47
1513	5.1	読み出し試験に用いるソフトウェアの概要	50
1514	5.2	RD53A シングルモジュール (SCC)	51
1515	5.3	モジュール付属サーミスタを用いた温度読み出し回路	52
1516	5.4	用いた電源	52
1517	5.5	使用した FPGA ボード (XpressK7)	53
1518	5.6	使用した FMC-DisplayPort 変換カード (オハイオカード)	53
1519	5.7	ハードウェアセットアップ	53
1520	5.8	ダウンロードしたモジュール ID 確認画面	54
1521	5.9	DCS 情報のモニタリングの様子	55
1522	5.10	検索機能確認の様子	56
1523	5.11	試験結果の閲覧	57
1524	5.12	読み出し試験結果の選択	59
1525	5.13	ピクセル解析結果	60
1526	5.14	各評価基準における不良ピクセルの分布	61
1527	5.15	中央データベースにおける読み出し試験及びピクセル解析結果画面	62
1528	6.1	検索機能実装方法 1:Python リストを用いた場合	65
1529	6.2	検索機能実装方法 1 の問題点	66
1530	6.3	検索処理時間のボトルネックとなっているデータ構造	66
1531	6.4	方法 1 における検索処理速度測定結果	67
1532	6.5	検索機能実装方法 2:検索キーワード専用コレクションを用いた場合	69
1533	6.6	方法 2 における検索処理時間測定結果	71
1534	6.7	方法 2 における検索情報コレクションの生成時間測定結果	72
1535	6.8	方法 2 の検索における詳細処理	73
1536	6.9	方法 2 における詳細処理時間の測定結果	74
1537	6.10	ドキュメント数に対する型変換処理時間の関係	75

1538	6.11	検索機能実装方法 3:検索情報と共に一覧表示に必要な情報を保持、参照	76
1539	6.12	検索機能実装方法 4:検索情報コレクションを分散、マルチスレッドを使用	76
1540	6.13	方法 3、4 に対する処理時間測定結果	77
1541	7.1	各サーバーの設置場所	79
1542	7.2	”createTestRunAttachment” を用いた添付処理におけるファイル容量と処理時間の関係	80
1543	7.3	ダウンロード処理における流れのイメージ図	83
1544	7.4	登録した Quad モジュールと構成部品のシリアルナンバー一覧。	84
1545	7.5	登録した Quad モジュールのダウンロードの様子	85
1546	7.6	モジュール及び構成部品情報取得のイメージ図	86
1547	7.7	中央データベースにおける読み出し試験結果の構造	89
1548	7.8	アップロード処理に関する流れのイメージ図	90
1549	7.9	アップロード機能における詳細処理測定結果	91
1550	8.1	中央データベースにおける選択可能な組み立て工程のイメージ	98
1551	A.1	真性半導体中のシリコン	99
1552	A.2	pn 接合の電流 – 電圧特性	100
1553	A.3	空乏層	100
1554	B.1	アナログフロントエンド	102
1555	B.2	デジタルフロントエンド	103
1556	B.3	フレキシブル基板	103
1557	C.1	ローカルデータベースシステムチュートリアルの様子	104
1558	C.2	ローカルデータベースシステム導入及び試運転場所	105
1559	D.1	生産時のモジュール組み立て状況解析の例	106
1560	E.1	添付するファイルサイズと処理時間の関係	108
1561	E.2	KEK、LBL 間のファイル送信	108
1562	E.3	KEK、LBL と CERN 間のファイル送信	109

表目次

1563	1.1	現行 LHC と HL-LHC の比較	6
1564	1.2	ピクセル検出器設置領域の比較	8
1565	1.3	搭載するピクセルモジュール数の比較	8
1566	1.4	VBF $H \rightarrow WW$ の測定における系統誤差の見積もり	9
1567	2.1	RD53A のスペック	12
1568	3.1	ピクセル解析の評価基準一覧	25
1569	4.1	品質試験に用いる主なコレクション	32
1570	4.2	中央データベースにおける組み立て工程と付随するテスト項目	36
1571	4.3	ローカルデータベースユーザ権限及び使用機能一覧	39
1572	5.1	温度読み出しシステムに使用した装置一覧	52
1573	5.2	読み出しに使用した PC の性能	53
1574	6.1	測定に使用したノート PC の性能	70
1575	6.2	検索機能処理時間測定の詳細	70
1576	6.3	処理 3,5 における型変換処理 6.5 の割合	74
1577	7.1	各ローカルデータベースサーバーの性能一覧	78
1578	7.2	同期ツールの中で使用する中央データベースの主な API 一覧	79
1579	7.3	中央データベース API 実行時の処理時間測定結果。	80
1580	7.4	ダウンロード機能を用いて保存する情報一覧。	81
1581	7.5	登録したモジュールのダウンロード処理時間測定結果	82
1582	7.6	ダウンロード機能における詳細処理にかかる時間測定	84
1583	7.7	中央データベースにおける読み出し試験結果に関する情報一覧	89
1584	7.8	アップロード処理における結果、設定値ファイル添付実行結果と処理時間	92
1585	7.8	アップロード処理における結果、設定値ファイル添付実行結果と処理時間	93
1586	7.9	アップロード処理改善後における結果、設定値ファイル添付実行結果と処理時間	94
1587			