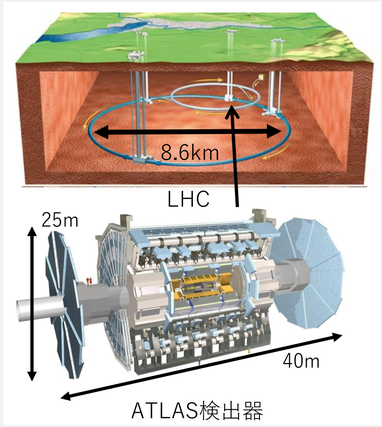
FPGAアクセラレータを用いた新しいファームウェア論理回路検証手法の開発

ICEPPシンポジウム　神戸大学 M1 水引 龍吾

1. ATLAS実験

高エネルギーの陽子を衝突させて新粒子や新しい物理を探索



LHCはジュネーブ (スイス)のCERNにある大型加速器
- 直径8.6km、全長27km
- 陽子を光速の99.999999%まで加速
- 毎秒4000万回、1回あたり平均50個衝突

正面衝突した陽子は様々な粒子を生成して飛び散る
高輝度化アップデートを控えており、**更にデータが増加**する

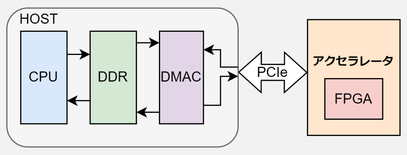
	稼働時期	エネルギー[TeV]	総データ量[fb^{-1}]
Run1	2011	7	5
	2012	8	21
Run2	2015-2018	13	139
Run3(現在)	2022-2025	13.6	250
HL-LHC	2029-2038	14	3000

2. FPGAとFPGAアクセラレータ

FPGA
- FPGAとは、内部回路を自由に書き換えできるデバイス
- HDL(ハードウェア記述言語)により内部回路を実装する
- 使用言語はverilog, system verilog


FPGA XCVU13P

FPGAアクセラレータ
- FPGAを内蔵し、演算を電気回路で高速化できる
- CPUと通信し、単一プラットフォームでFPGA実装可能



PCとの接続
- PCIeで接続
- 通信プロトコルはAXI4
- 最大バースト長256bit

FPGAアクセラレータ Alveo U200

3.トリガーシステム

検出器の信号を元に
データを取捨選択するシステム

陽子衝突

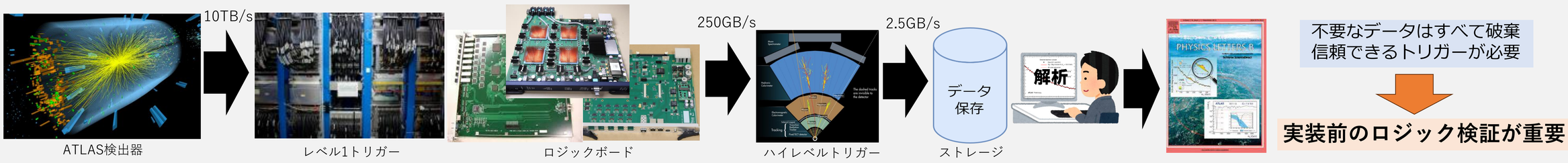
ATLAS検出器は1秒間に10TBのデータが発生。レベル1トリガーでデータを1/400に削減

ハードウェアを用いて高速に判定。 ボードにはFPGAを搭載

保存されたデータを用いて詳細に解析

ソフトウェアで精密に判定。選ばれたデータを保存

解析して新しい物理事象を発見する



ATLAS検出器　レベル1トリガー　ロジックボード　ハイレベルトリガー　ストレージ

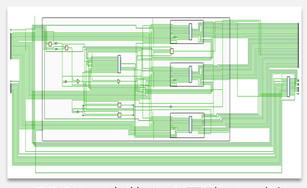
不要なデータはすべて破棄
信頼できるトリガーが必要

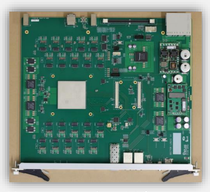
実装前のロジック検証が重要

4. システム開発のモチベーション

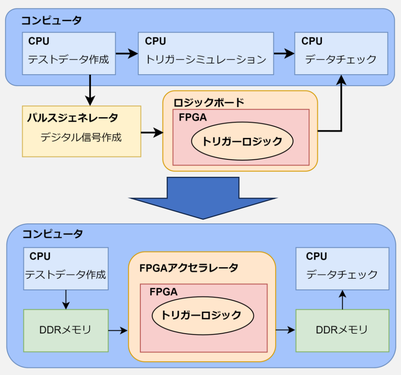
今のファームウェア検証システムが持つ課題
- トリガーロジックは電気回路としてFPGAに実装
- 書き込む回路は年々**大規模化、複雑化**
- ボードは高価で数も少ないため、検証難易度が高い

→FPGAアクセラレータで解決できないか？


FPGAに実装する回路の一例


実験で使用するボード(Sector Logic)

プラットフォームの単一化



コストの削減

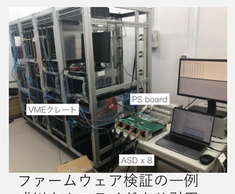
FPGA	XCVU13P (実験で使用)	FPGAアクセラレータ Alveo U200	Alveo U250
値段	$ 59,000	$ 6,000	$ 9,300
LUTs	1,728 k	1,182k	1,728k
BRAM	95768 kb	77760 kb	95768 kb
URAM	368,640 kb	276,480 kb	368,640 kb

- アクセラレータは大量生産され、安価で入手可能
- 1/10の価格で同等のスペック
- バスの少なさが唯一の欠点

場所の制約
- 現在のシステムでは物理的配線が必要
- 専用の設備とノウハウが必要

→ボードがある施設に移動しないといけない

- 検証システムがアクセラレータカード内で完結すればネットワーク上で実行可能になる

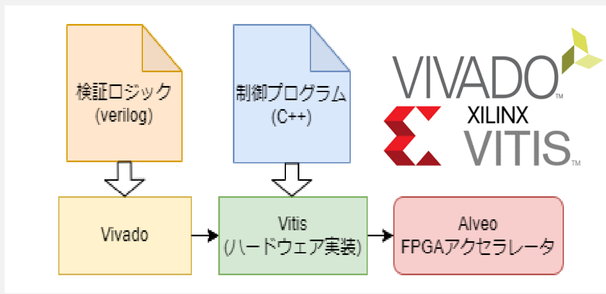


ファームウェア検証の一例
成川さんスライドより引用

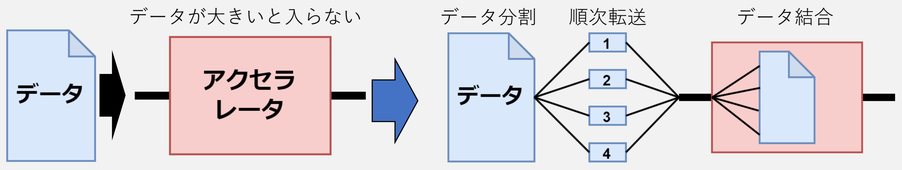

大学のPCで運用している
Alveoアクセラレータカード

5.ファームウェア検証システムの開発

開発手法
- Xilinx社が提供するVivadoツールとVitisツールで開発
- Vivadoでファームウェアの実装。
- VitisでFPGAに回路実装を行う。

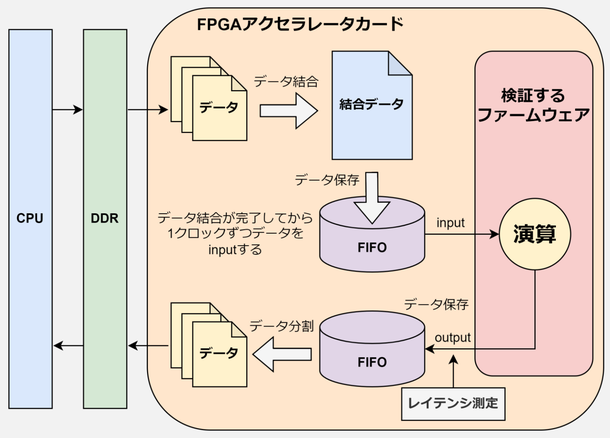


課題点と解決
- FPGAアクセラレータはバスが少なく、大量のデータ転送に不向き
- そのままでは検証システムとして使えない
- データの結合とパイプラインを実装して問題を解決



レイテンシ測定機能
- クロック信号を管理することでレイテンシ測定機能を実装
- データの入力から出力までの所要時間が測定できる

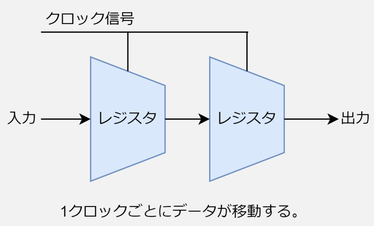
開発したシステムのブロック図



6. 開発したシステムの検証

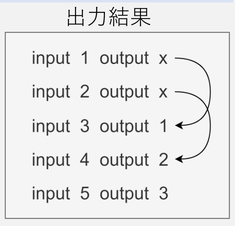
レイテンシ測定

ファームウェアの遅延を測定することができる



1クロックごとにデータが移動する。

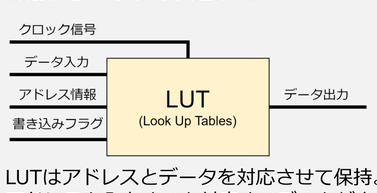
出力結果



2クロックのズレを確認

LUT検証

実際の実験で使用するLUT(Look Up Tables)に乱数を入力して検証した



LUTはアドレスとデータを対応させて保持。
アドレスを入力すると対応するデータが出力

アドレス	入力データ
0	9
1	11
2	14
3	4
4	

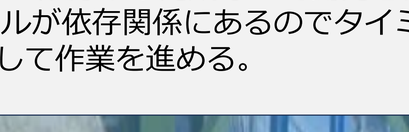
出力結果

address: 0 output: 9
address: 1 output: 11
address: 2 output: 14
address: 3 output: 4

対応するデータが
正しいことを確認

大規模入力検証

巨大な入出力を持つファームウェアを検証



N + N = 2Nのロジック

出力結果

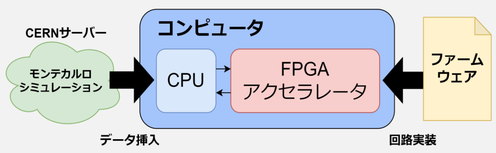
output 0
output 2
output 4
output 6
output 8

問題なく動作を確認

7. 今後の研究方針

○ シミュレーションデータの使用

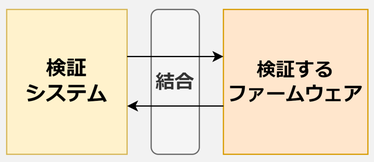
シミュレーションを用いた検証プロセス



- LUTに欠陥があると適切な物理事象のデータを破棄してしまう
- シミュレーションデータでモジュールに使われるLUTの検証をする

○ プロセス自動化

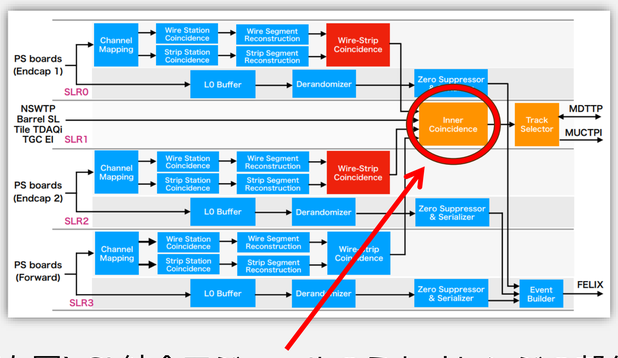
検証システム-ファームウェア間の結合



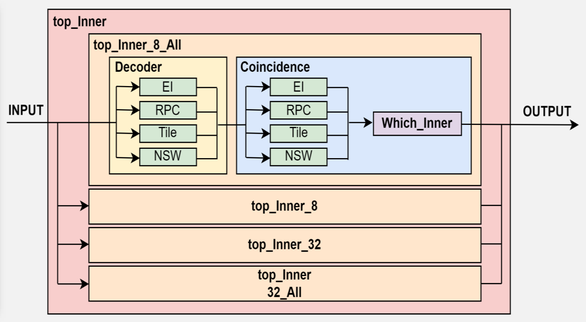
- 検証システムは汎用性が高いが、検証したいファームウェアとの結合は手作業
- 自動化で検証プロセスの簡易化を目指す

○ HL-LHCのために開発中のトリガーロジックのデバッグ作業、統合作業

SL統合ファームウェア実装状況



Inner Coincidenceのモジュール依存関係



(左図) SL統合モジュールのうちオレンジの部分が未完成なので開発とデバッグを行う。
(右図) ファームウェアは複数のモジュールが依存関係にあるのでタイミング調節が必要。
今回開発した検証システムを応用して作業を進める。