



Flash Memory Summit

Database Caching and Query Acceleration using SmartSSD

Alex Paek

AMD
















Agenda

- Greenplum Database (GPDB)
- Samsung SmartSSD CSD
- Accelerating GPDB using SmartSSD
- Acceleration Library

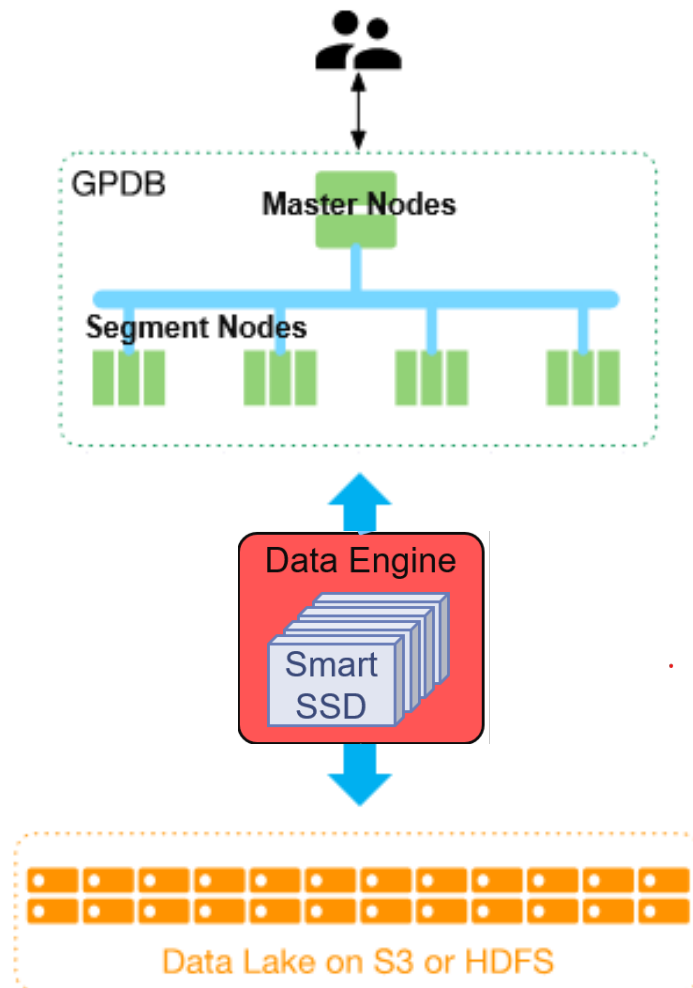
Database Management SW (DBMS) Ranking

395 systems in ranking, July 2022

Rank	Rank			DBMS	Database Model	Score		
	Jul 2022	Jun 2022	Jul 2021			Jul 2022	Jun 2022	Jul 2021
1.	1.	1.		Oracle +	Relational, Multi-model 	1280.30	-7.44	+17.63
2.	2.	2.		MySQL +	Relational, Multi-model 	1194.87	+5.66	-33.51
3.	3.	3.		Microsoft SQL Server +	Relational, Multi-model 	942.13	+8.30	-39.83
4.	4.	4.		PostgreSQL +	Relational, Multi-model 	615.87	-4.97	+38.72
5.	5.	5.		MongoDB +	Document, Multi-model 	472.98	-7.74	-23.18
6.	6.	6.		Redis +	Key-value, Multi-model 	173.62	-1.69	+5.32
7.	7.	7.		IBM Db2	Relational, Multi-model 	161.22	+2.03	-3.94
8.	8.	8.		Elasticsearch	Search engine, Multi-model 	154.33	-1.67	-1.43
9.	9.	↑ 11.		Microsoft Access	Relational	145.09	+3.27	+31.64
10.	10.	↓ 9.		SQLite +	Relational	136.68	+1.24	+6.47
11.	11.	↓ 10.		Cassandra +	Wide column	114.40	-1.05	+0.40
12.	12.	12.		MariaDB +	Relational, Multi-model 	112.52	+0.94	+14.54
13.	13.	↑ 25.		Snowflake +	Relational	99.15	+2.73	+59.11
14.	14.	↓ 13.		Splunk	Search engine	98.21	+2.64	+8.15
15.	15.	15.		Microsoft Azure SQL Database	Relational, Multi-model 	84.89	-1.12	+9.68
16.	16.	16.		Amazon DynamoDB +	Multi-model 	83.94	+0.05	+8.74
17.	17.	↓ 14.		Hive +	Relational	79.48	-2.10	-3.19
18.	18.	↓ 17.		Teradata +	Relational, Multi-model 	70.93	+0.52	+1.98
19.	19.	↓ 18.		Neo4j +	Graph	58.42	-1.11	+1.25
20.	20.	20.		Solr	Search engine, Multi-model 	55.70	-0.92	+3.90

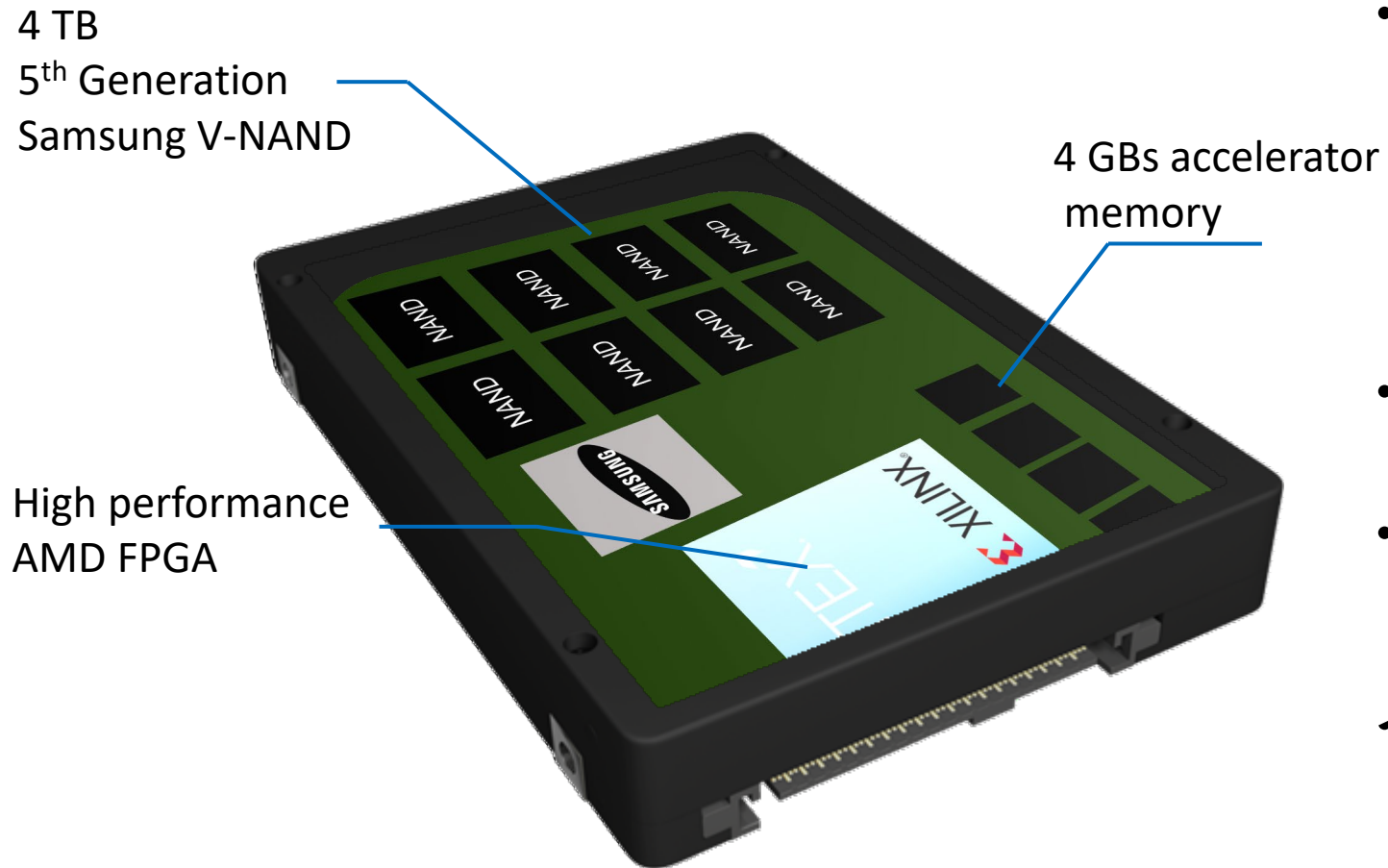
<https://db-engines.com/en/ranking>

Greenplum Database (GPDB)



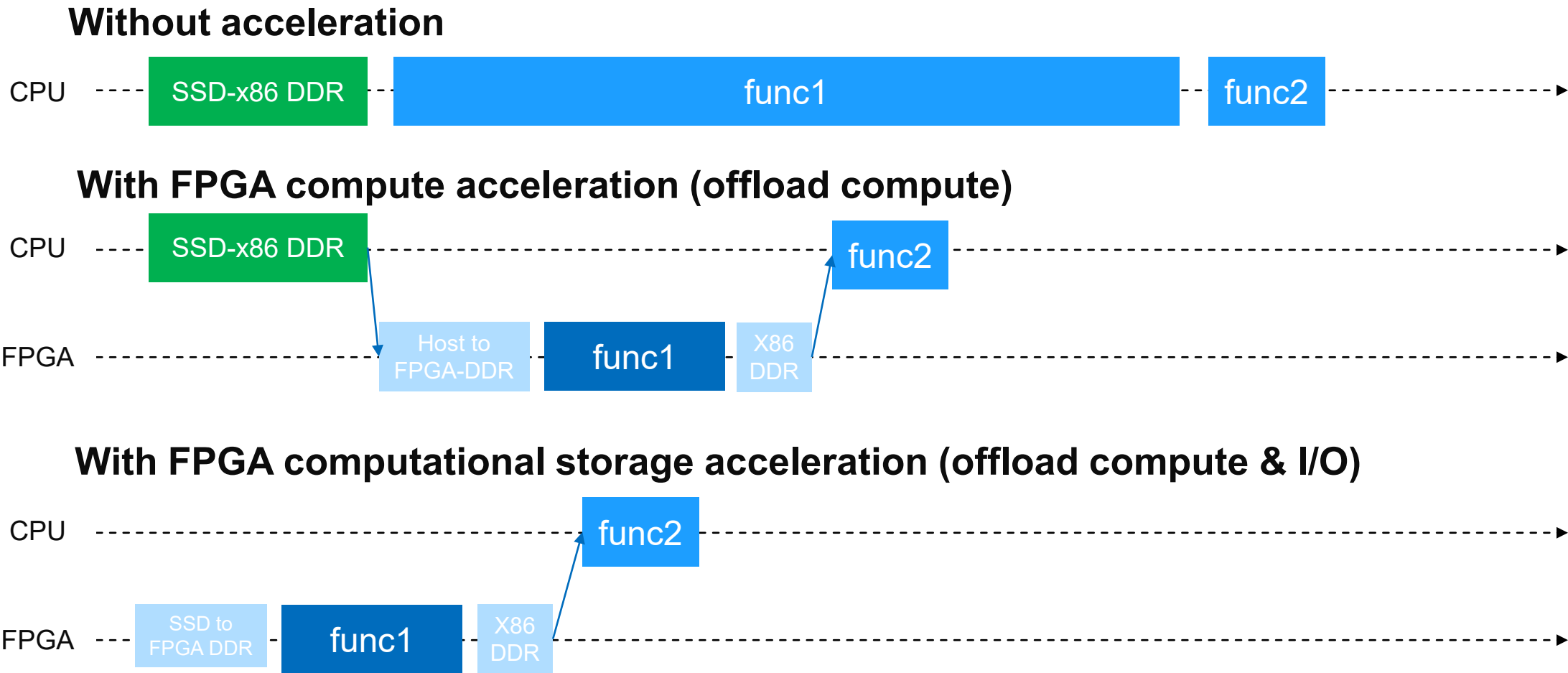
- Based on PostgreSQL and MPP (massively parallel processing) architecture
- GPDB cluster consists of a master node, segment nodes
- All the data must be loaded to segment nodes from data lake before query can start – time consuming process
- **Data Engine, utilizing Samsung SmartSSD**
 - Caches data from data lake
 - Eliminates the data loading. Query can start right away
 - Process subset of query from segment nodes
 - FPGA acceleration - decompression, parsing, filter, aggregate, group by function
 - The result data to GPDB segments are much smaller

Samsung SmartSSD CSD



- Provides storage and acceleration capabilities on one device (Computational Storage Device or CSD)
 - Storage: Samsung SSD = controller ASIC + NAND Flash
 - Acceleration: Xilinx KU15P FPGA
- U.2 Form Factor
 - PCIe Gen 3x4 Interface
- Provides CPU offload capability
 - Overcomes CPU-based limitations on throughput, latency, scaling, and data access
- Supports Vitis/HDK design flow

Computational Storage Benefit

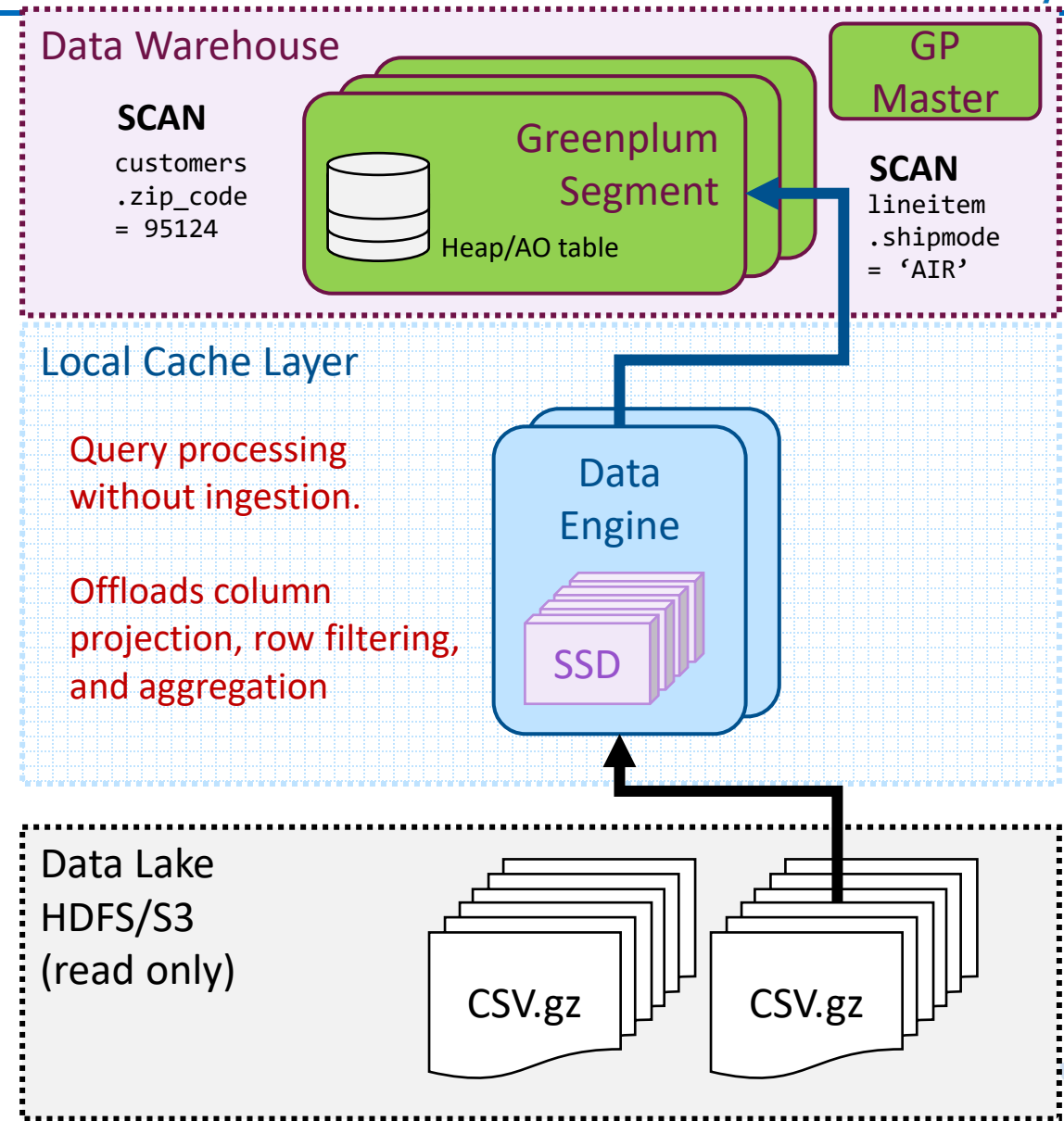


Computational Storage Solution avoids copying to x86 DDR



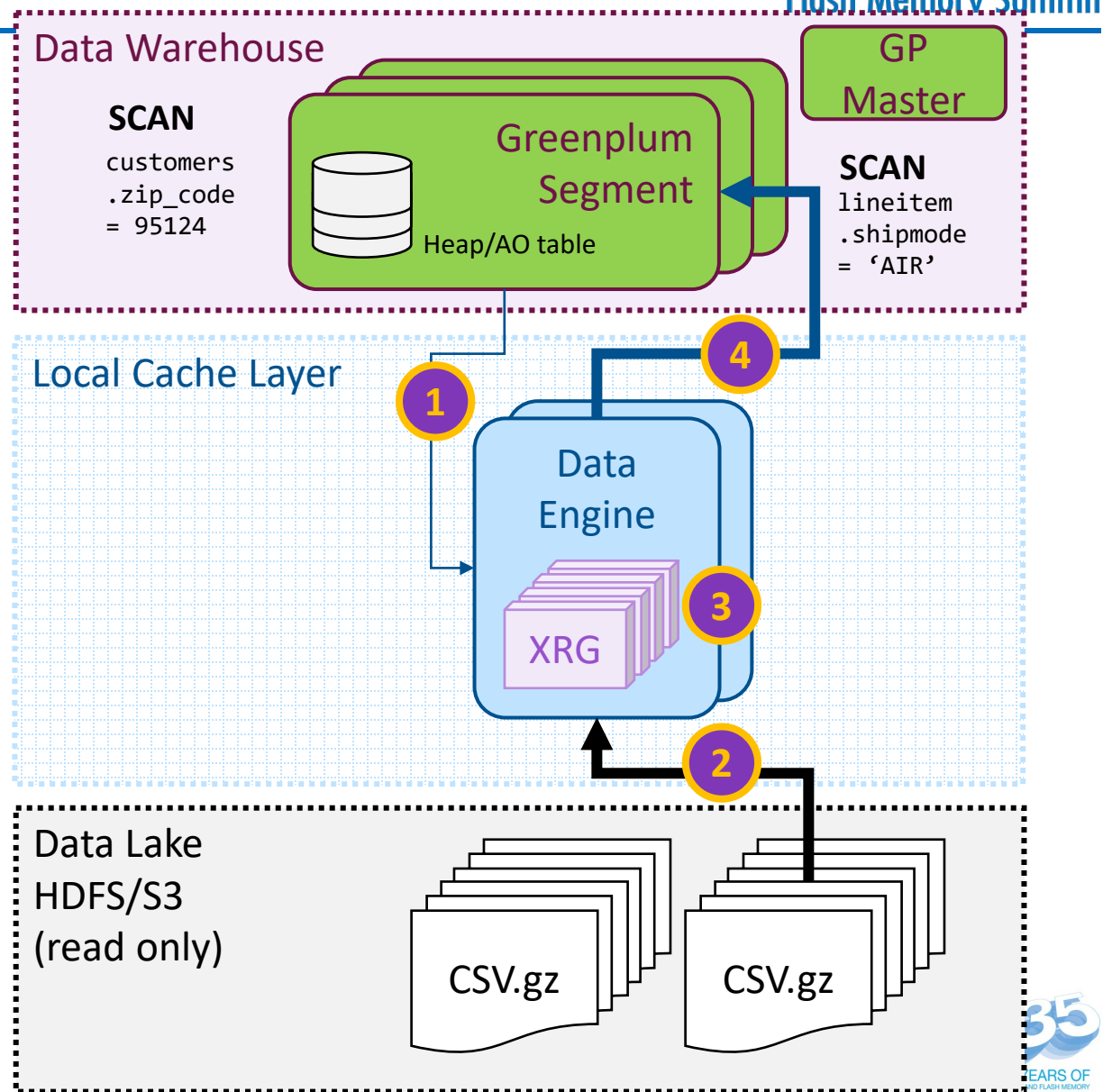
Empower Greenplum with SmartSSD

- **Internal table**
 - Small, *dimensional* data
 - Changed frequently (hard to track latest data)
- **External table**
 - Large, *fact* data
 - High real-world demand for query without ingestion (loading into data warehouse)
- **Data Engine**
 - Enables Greenplum DWH to efficiently scan files from data lake on demand as external table, by
 - Uses SmartSSD as cache
 - Executes pushdowns



Data Engine Workflow

1. GP issues external table request
2. Data Engine fetches and lands HDFS/S3 data as XRG files if data is not available in cache
 - XRG is binary format that enables zone-map based filtering and easy FPGA acceleration
3. Data Engine scans XRG files with SmartSSD FPGA acceleration
4. Data Engine serves filtered data to GP segments



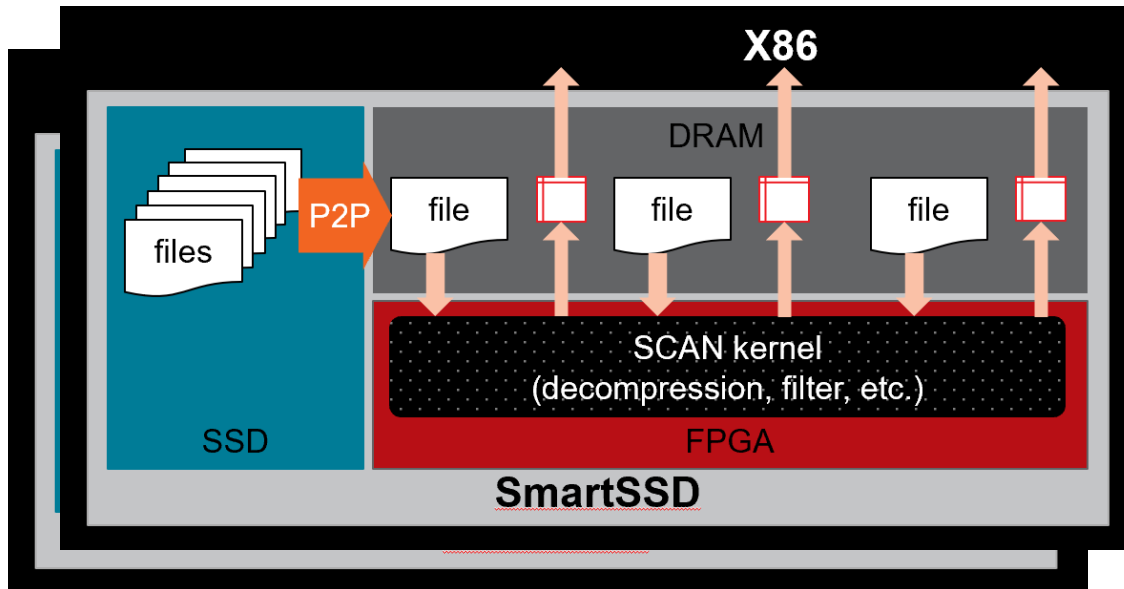


Accelerating SCAN Operation via SmartSSD

- Local PCIE P2P

- Raw data DMA'd from SSD to FPGA DRAM within SmartSSD, bypassing CPU

- Parallel processing in multiple SmartSSD drives



- FPGA accelerated operations

- Decompression
- Row filtering
 - Native types like INT8, INT32, INT64, FLOAT, DOUBLE
 - Database specific types like DECIMAL, DATE, TIMESTAMP
 - Strings for equal, IN(candidate1, candidate2, ...), LIKE "%pattern%"

- FPGA processing "at line rate"

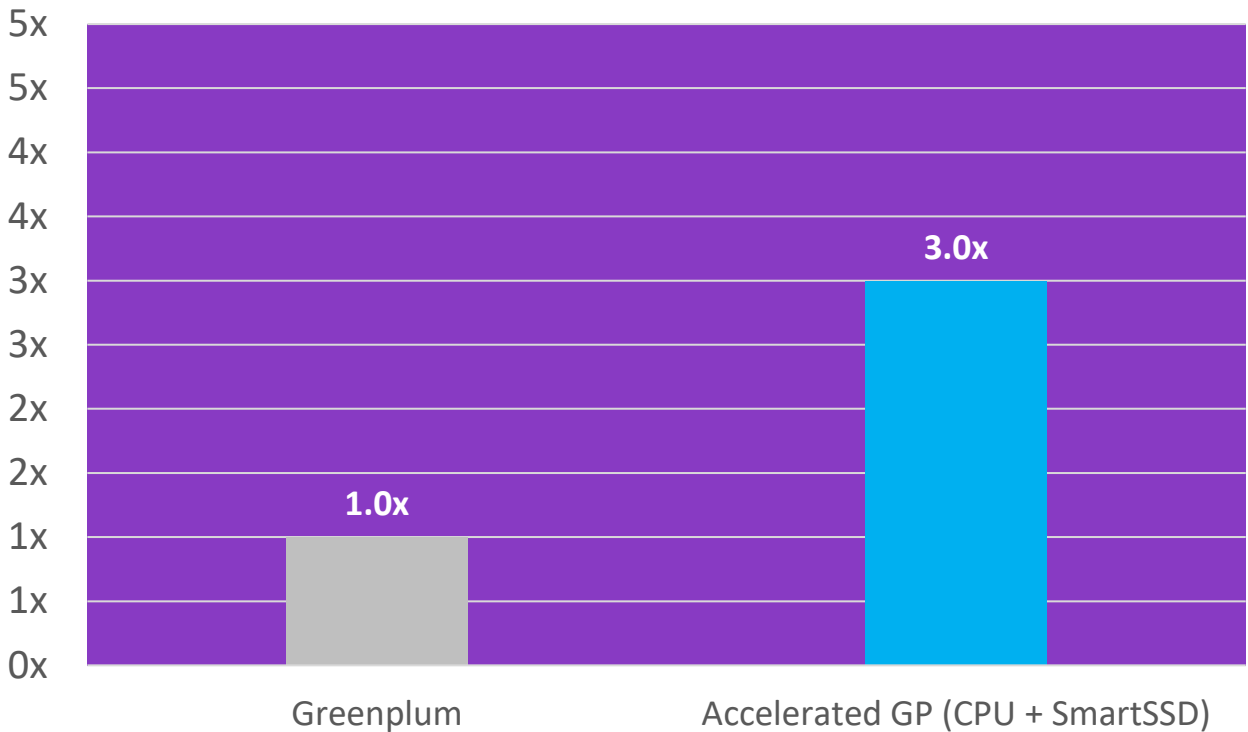
- E.g., query for shipmode = 'REG AIR'
 - 40 vCPU running Greenplum archives ~600MB/s
 - 1 SmartSSD's FPGA kernels can process over 4GB/s and it scales with number of drives
 - End-to-end performance bounded by P2P/ PCIE bandwidth of U.2

Game Changer: SmartSSD

- >3x performance gain
- Using TPC-H benchmark 30SF data
- 2 servers – 1 for GPDB master/segment, 1 for Data Engine
- Server configuration
 - AMD7313, 16C/32C. 512 GB memory, NVMe SSD
 - 2x25G Mellanox NIC



Projected Speed Up With SmartSSD



Vitis and Vitis Libraries

- Vitis offers a complete set tools to build, analyze performance bottlenecks and debug accelerated algorithms, developed in C, C++ or OpenCL.
- Vitis Libraries
 - Open Source
 - Commercial-friendly Apache 2.0 license
 - Published on GitHub
 - Performance-Optimized
 - 700+ APIs in multiple domains
 - Familiar Programming Language
 - All backbone IPs on FPGA or ACAP are written in C++
 - No Verilog or VHDL knowledge required to understand and use the APIs
 - Documented
 - In-code and HTML documentation for all APIs

Xilinx / Vitis_Libraries Public

<> Code Issues 38 Pull requests 4 Discussions Actions Projects

master 7 branches 12 tags Go to file Code

sdausr create master branch from next branch fb6af2d 8 days ago 116 commits

blas Squashed 'blas' changes from e1c3094..c3b742c (...) 20 days ago

codec add api meta josn to codec (#646) 9 days ago

data_analytics Squashed 'data_analytics' changes from 4f3b370..... 9 days ago

Vitis Data Analytics Library2022.1

Search docs

Library Overview

Vitis Data Analytics Library

Release Note

Requirements

Design Flows

Vitis Data Analytics Library Tutorial

L1 User Guide

Hardware Classes

Hardware Functions in xf::data_analytics::classification

Hardware Functions in xf::data_analytics::clustering

Hardware Functions in xf::data_analytics::...

#include "xf_data_analytics/text/editDistance.hpp"

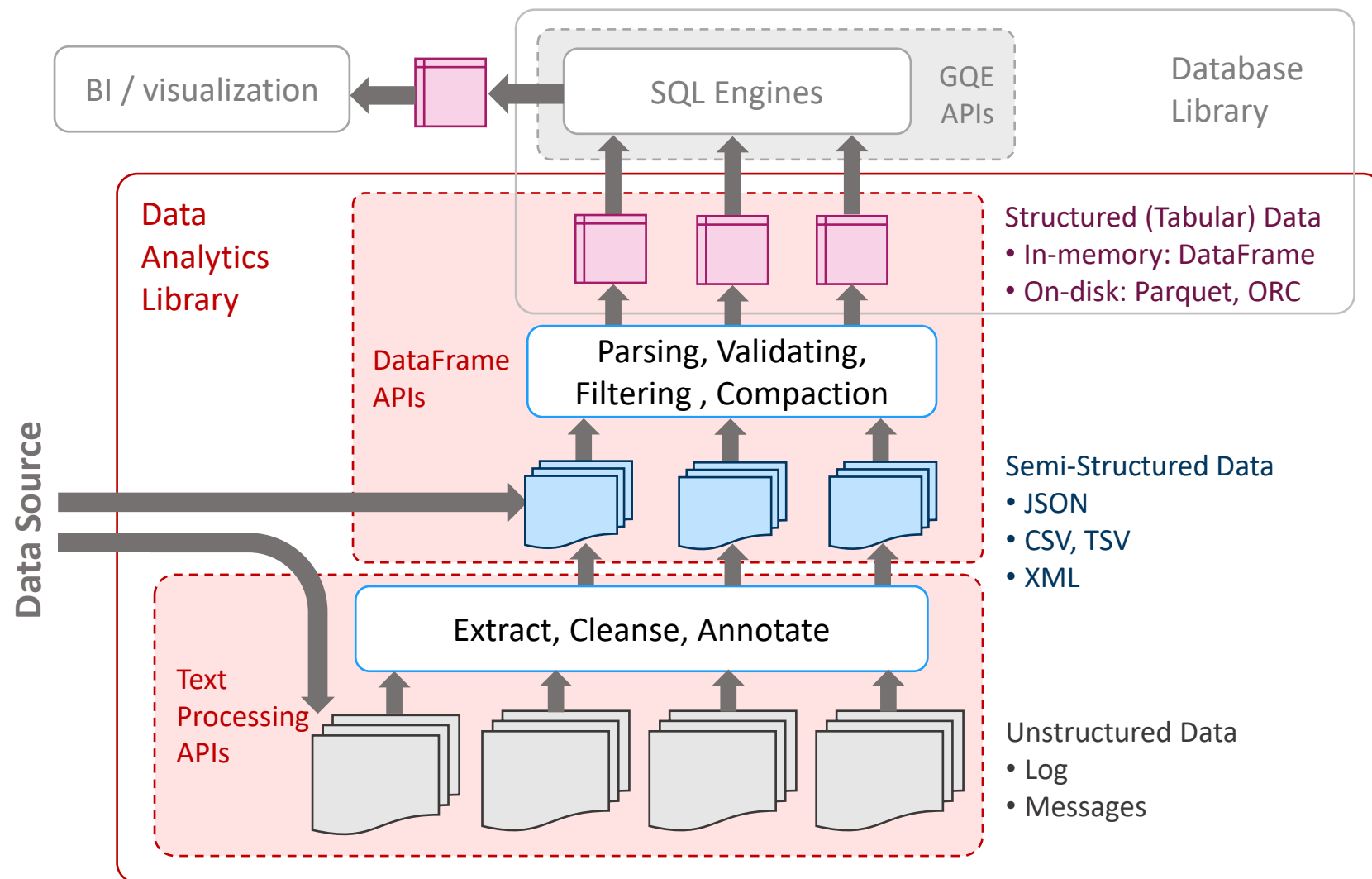
```
template <
    int N,
    int M,
    int BITS
>
void editDistance (
    hls::stream<ap_uint<BITS>>& len1_strm,
    hls::stream<ap_uint<64>>& query_strm,
    hls::stream<ap_uint<BITS>>& len2_strm,
    hls::stream<ap_uint<64>>& input_strm,
    hls::stream<ap_uint<BITS>>& max_ed_strm,
    hls::stream<bool>& i_e_strm,
    hls::stream<bool>& o_e_strm,
    hls::stream<bool>& o_match_strm
)
```

Levenshtein distance implementation.

Parameters:

N	maximum length of query string.
M	maximum length of input stream string, N must be less than M.
BITS	data width of internal edit distance in bits.

Database Library



• DataFrame

- Arrow read-write
- Parquet write
- JSON parse (POC)
- CSV parse (in-dev)

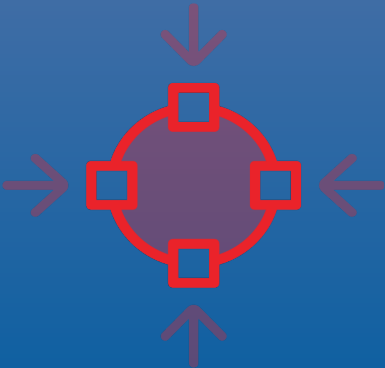
• Text Processing

- Regex match
- Geo-IP lookup
- Deduplication



SmartSSD Use Cases

Flash Memory Summit



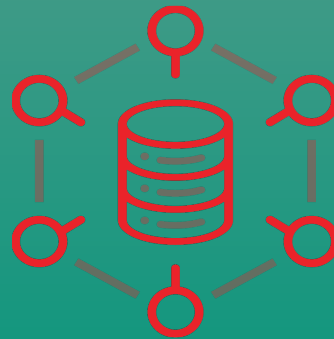
Better GB/\$

Transparent Compression



More # of Video Streams and better latency

Video File Transcoding



Higher Performance

Search in storage



Better TCO

Big Data Analytics (Database)

