SRC JUMP

CRISP
Center for Research on Intelligent
Storage and Processing in Memory

INSIDER: Designing In-Storage Computing System for Emerging High-Performance Drive

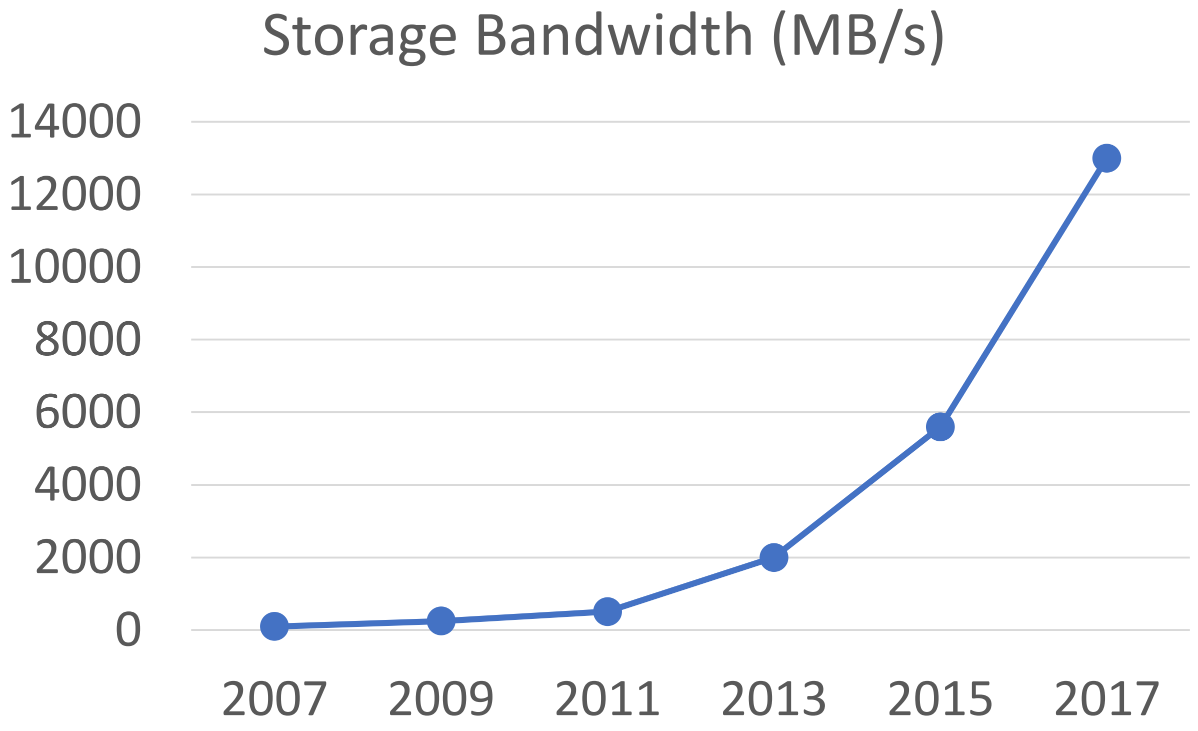
Zain (Zhenyuan) Ruan, Tong He, Jason Cong

University of California, Los Angeles

UCLA

Background: Data Movement Bottleneck

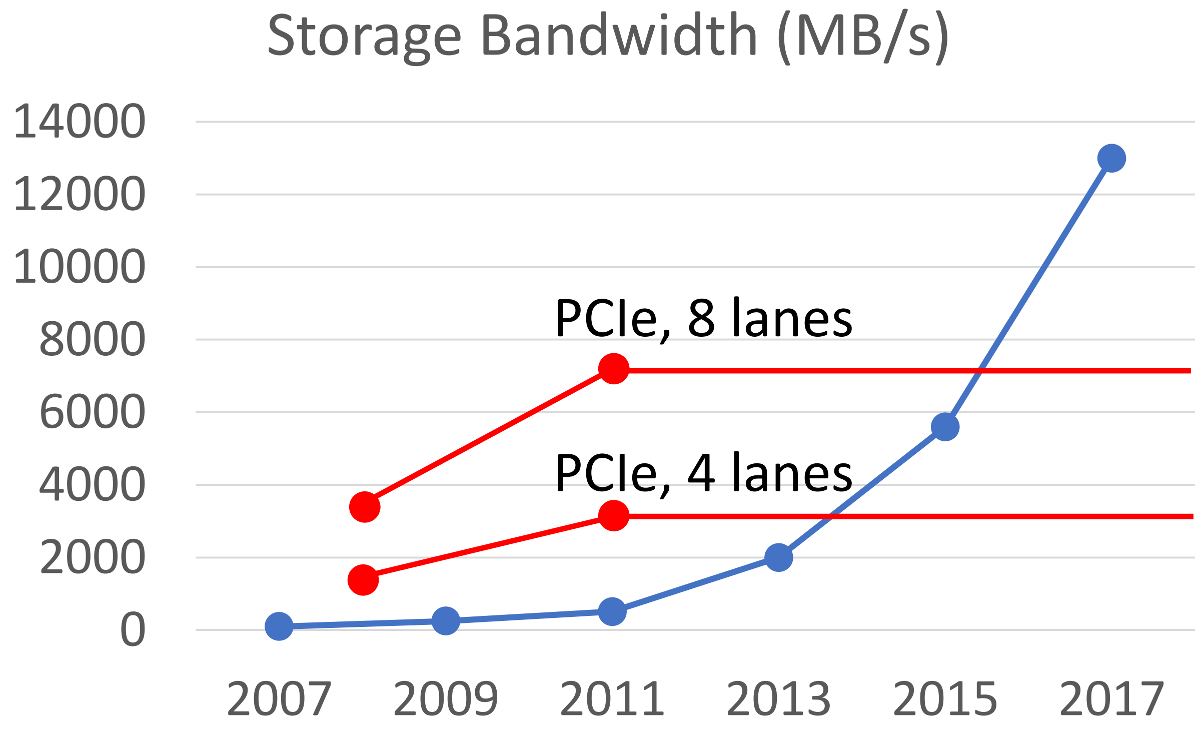
- "Moore's Law" of storage drive: bandwidth doubles every two years.



Background: Data Movement Bottleneck

- "Moore's Law" of storage drive: bandwidth doubles every two years.

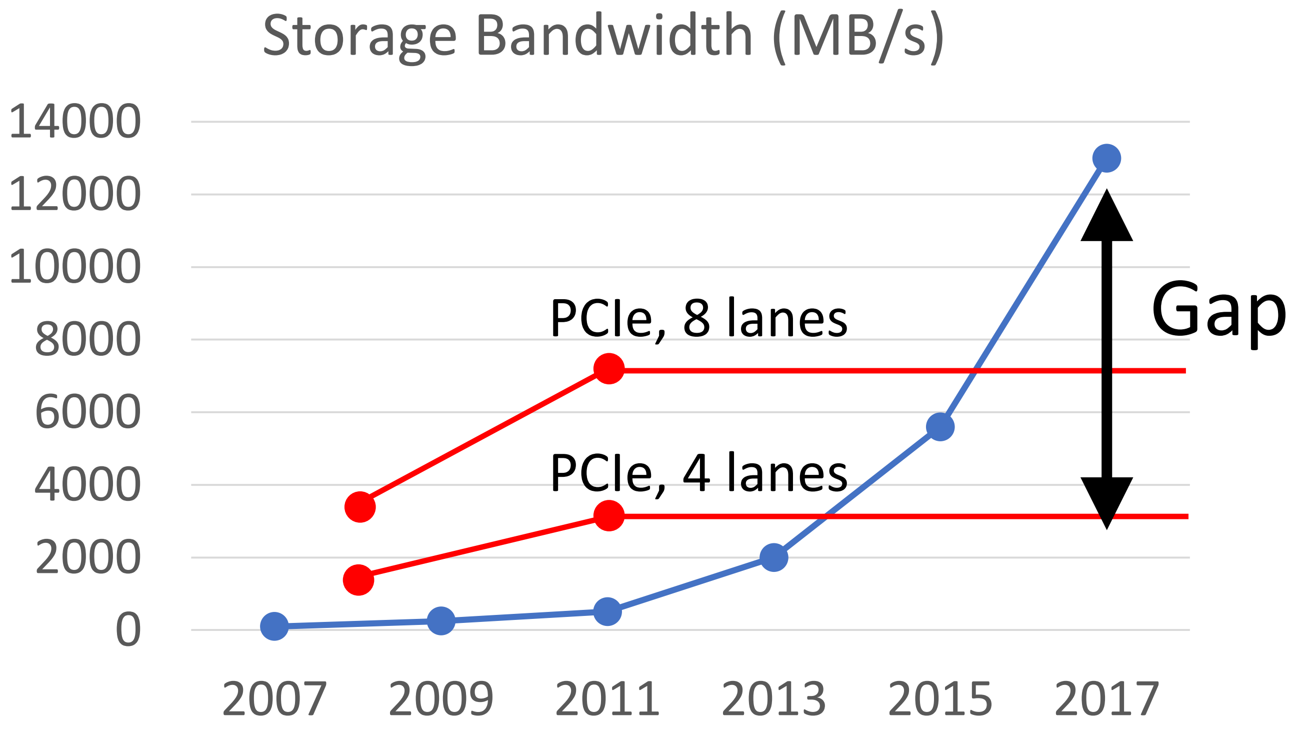
➢ The interconnection performance does not scale well.



Background: Data Movement Bottleneck

- "Moore's Law" of storage drive: bandwidth doubles every two years.

➢ The interconnection performance does not scale well.

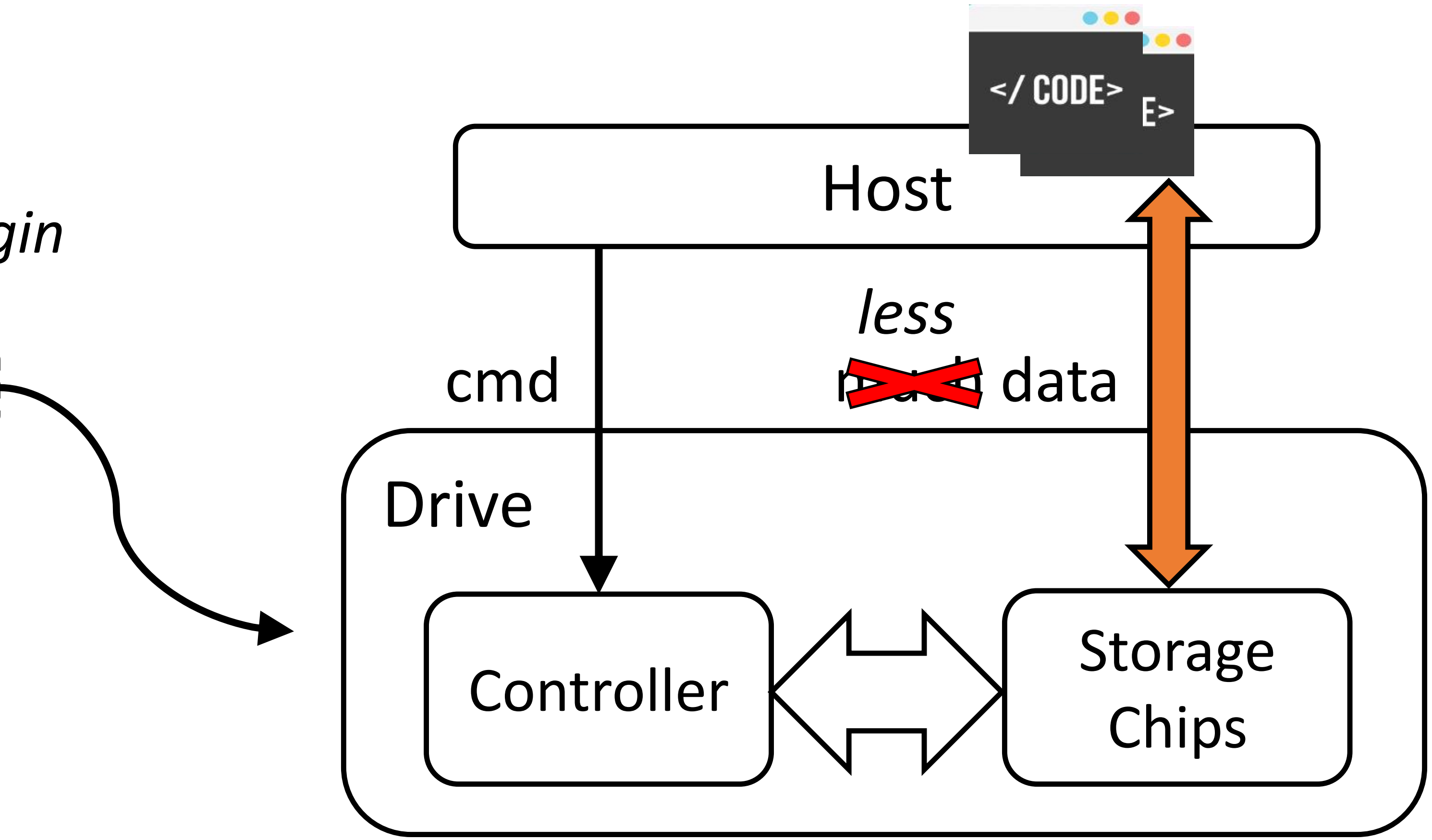


Existing Work

- In-storage computing (ISC).

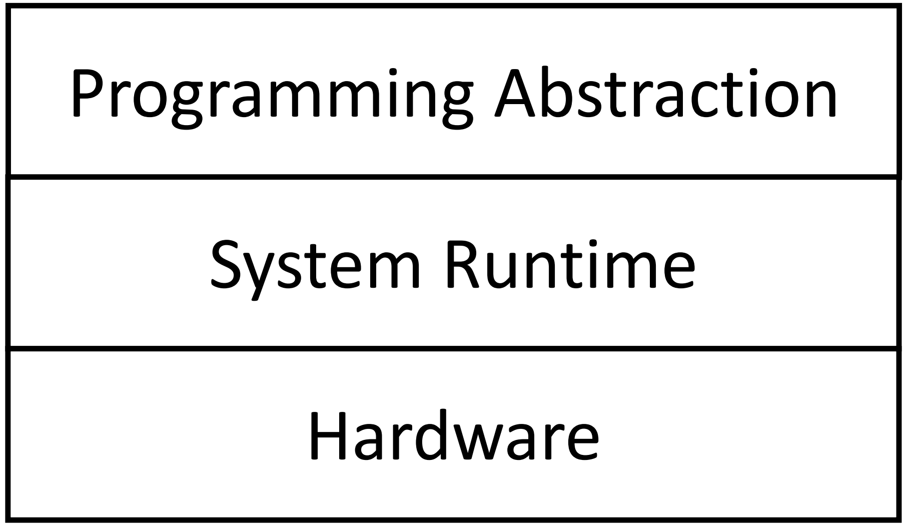
Example:

```
SELECT AVG(depdelay), origin
FROM flight_delays
WHERE distance > 2000
GROUP BY origin
ORDER BY flight_id;
```



Limitations of Existing Work

- Analyzing existing work by examining every layer of the system stack.



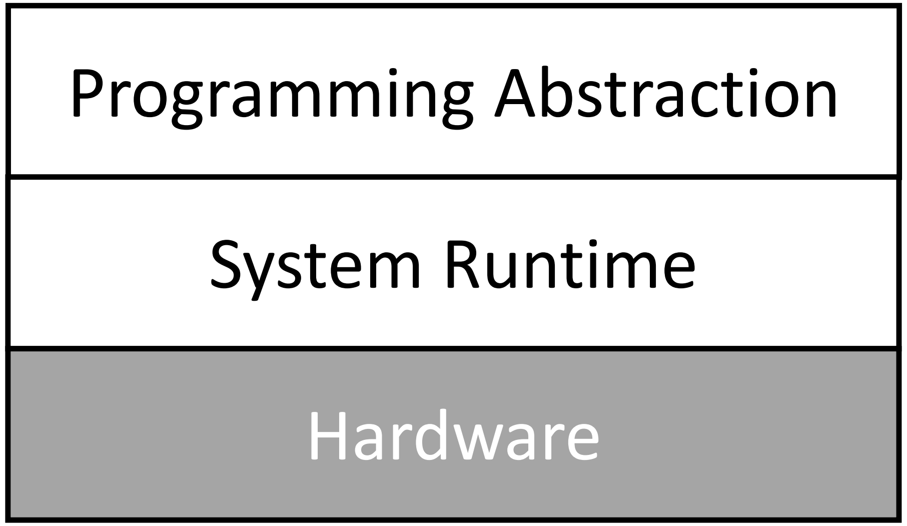
Programming Abstraction

System Runtime

Hardware

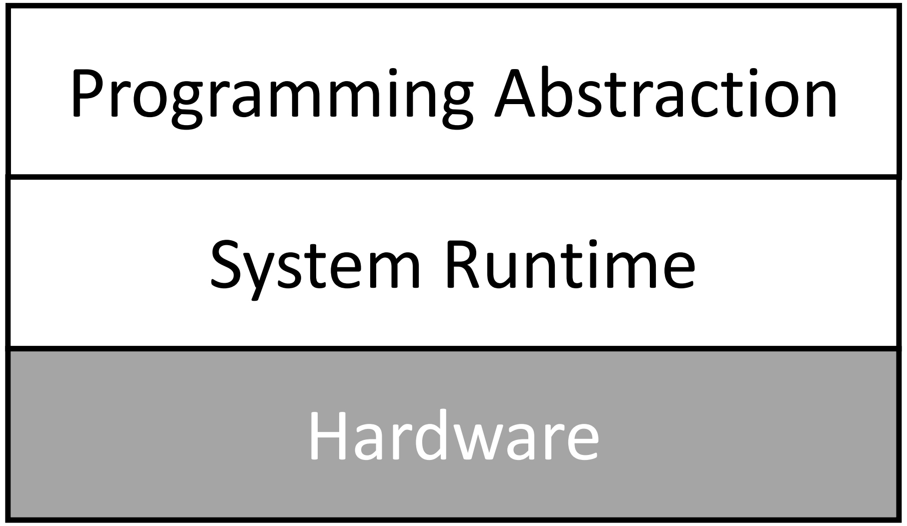
Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.



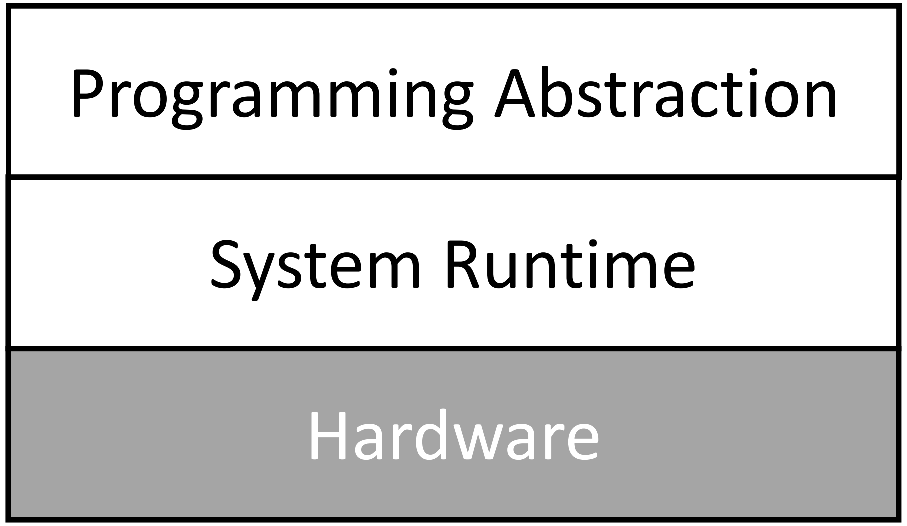
Limitations of Existing Work

- HW: Limited performance or flexibility.
    - ARM-based --- insufficient computing speed.
    - ASIC-based --- limited to specific apps.



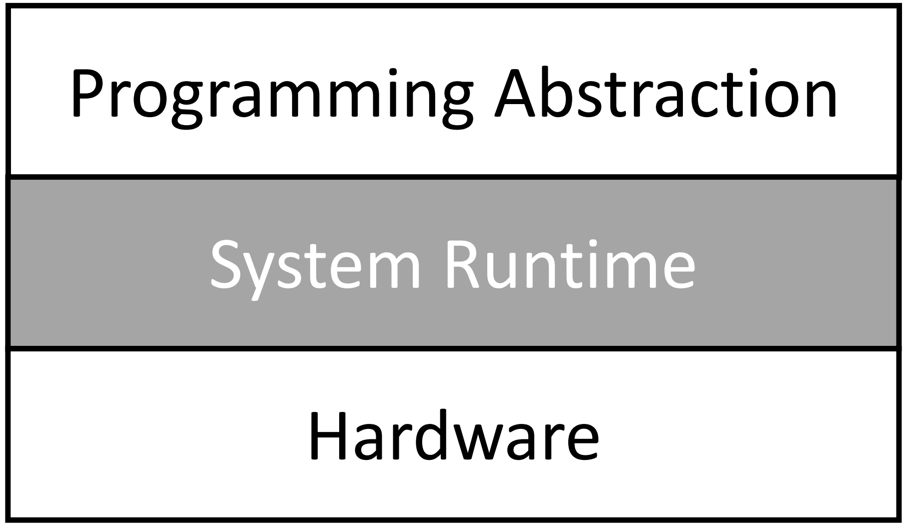
Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ➢ASIC-based --- limited to specific apps.



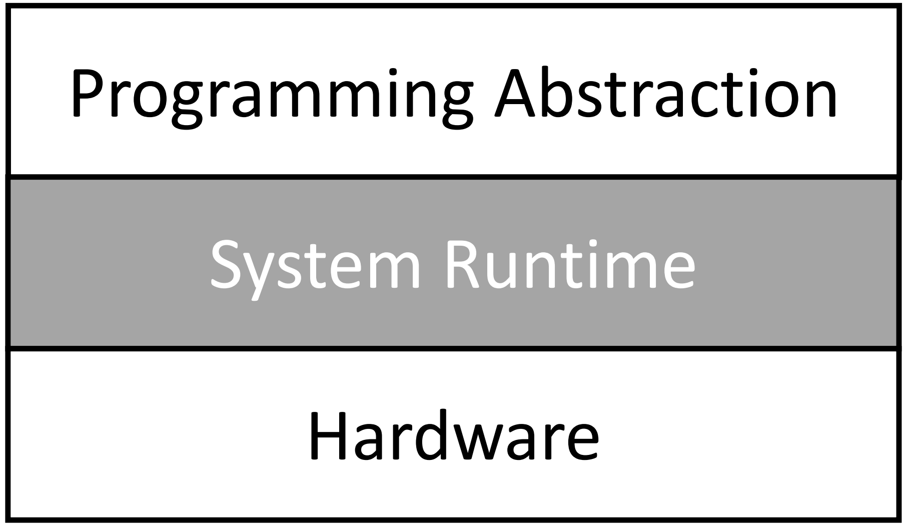
Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- ➢Runtime: Lack of crucial supports.
  - Protection:
    Drive prog. may access unwarranted data.
  - Virtualization:
    Support simultaneous multiple drive progs.



Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- Runtime: Lack of crucial supports.
  - ➢Protection:
    Drive prog. may access unwarranted data.
  - Virtualization:
    Support simultaneous multiple drive progs.

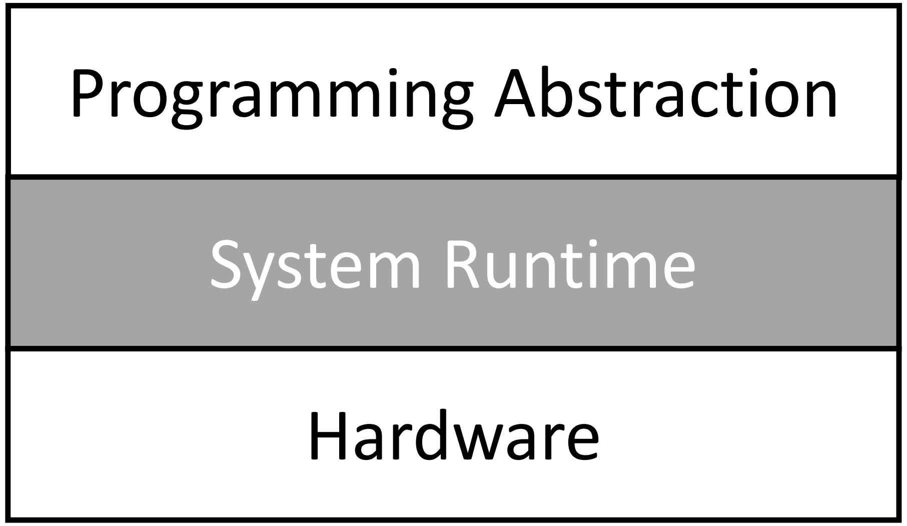


Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- Runtime: Lack of crucial supports.
  - Protection:

    Drive prog. may access unwarranted data.
  - Virtualization:

    Support simultaneous multiple drive progs.



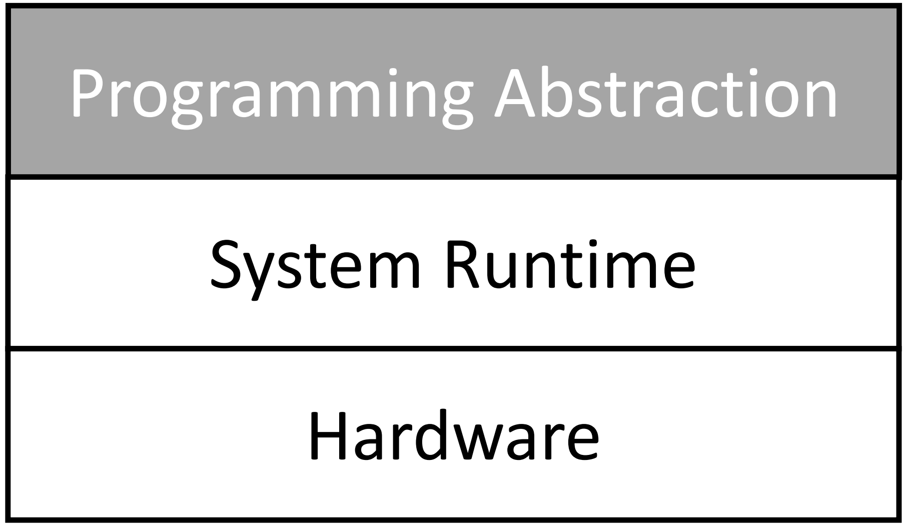
Programming Abstraction

System Runtime

Hardware

Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- Runtime: Lack of crucial supports.
  - Protection:
    Drive prog. may access unwarranted data.
  - Virtualization:
    Support simultaneous multiple drive progs.
- Programming: Lack of a simple abstraction.
  - Not integrated with the existing system interface.
  - Require considerable code modifications.



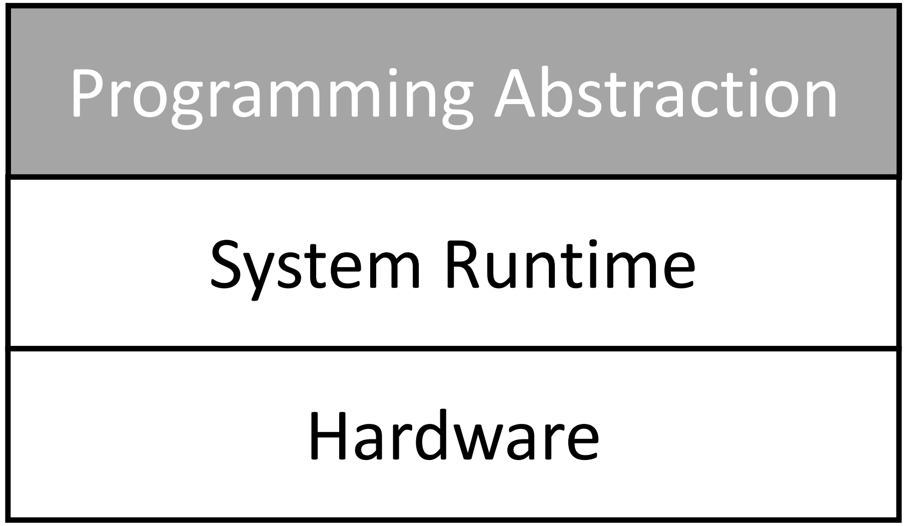
Programming Abstraction

System Runtime

Hardware

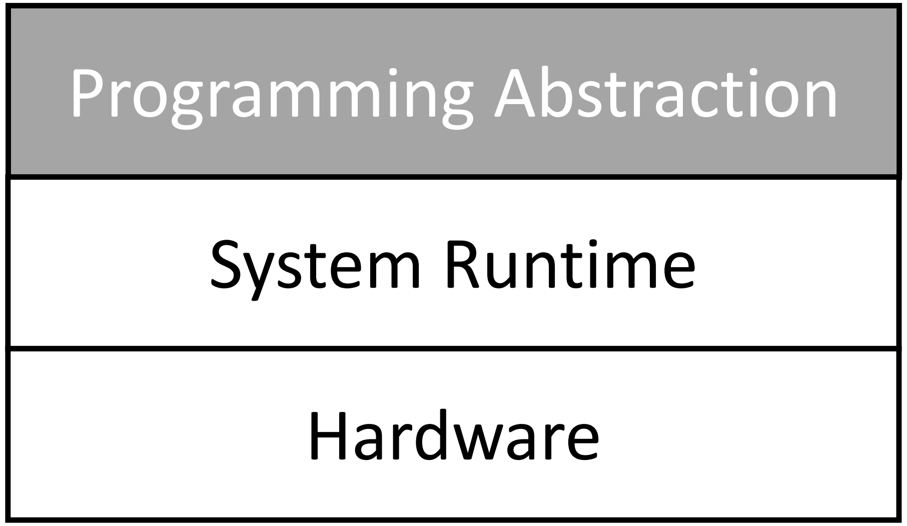
Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- Runtime: Lack of crucial supports.
  - Protection:
    Drive prog. may access unwarranted data.
  - Virtualization:
    Support simultaneous multiple drive progs.
- Programming: Lack of a simple abstraction.
  - Not integrated with the existing system interface.
  - Require considerable code modifications.



Limitations of Existing Work

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- Runtime: Lack of crucial supports.
  - Protection:
    Drive prog. may access unwarranted data.
  - Virtualization:
    Support simultaneous multiple drive progs.
- Programming: Lack of a simple abstraction.
  - Not integrated with the existing system interface.
  - ➢Require considerable code modifications.



INSIDER’s Approach

- HW: Limited performance or flexibility.
  - ARM-based --- insufficient computing speed.
  - ASIC-based --- limited to specific apps.
- Runtime: Lack of crucial supports.
  - Protection:

    Drive prog. may access unwarranted data.
  - Virtualization:

    Support simultaneous multiple drive progs.
- Programming: Lack of a simple abstraction.
  - Not integrated with the existing system interface.
  - Requires considerable code modifications.

INSIDER's Approach

- ~~HW: Limited performance or flexibility.~~
  - ~~ARM-based --- insufficient computing speed.~~
  - ~~ASIC-based --- limited to specific apps.~~

➢ HW: FPGA-based.
12X perf., 31X cost efficiency.

- Runtime: Lack of crucial supports.
  - Protection:
    Drive prog. may access unwarranted data.
  - Virtualization:
    Support simultaneous multiple drive progs.
- Programming: Lack of a simple abstraction.
  - Not integrated with the existing system interface.

  ➢ Requires considerable code modifications.

INSIDER’s Approach

- ~~HW: Limited performance or flexibility.~~
  - ~~ARM-based --- insufficient computing speed.~~
  - ~~ASIC-based --- limited to specific apps.~~

- HW: FPGA-based.

12X perf., 31X cost efficiency.

- Runtime: Lack of crucial supports.
  - ~~Protection:~~

    ~~Drive prog. may access unwarranted data.~~
  - Virtualization:

    Support simultaneous multiple drive progs.

➢ Runtime:

A separate control plane that enforces permission check and resource scheduling.

- Programming: Lack of a simple abstraction.
  - Not integrated with the existing system interface.

  ➢ Requires considerable code modifications.

INSIDER's Approach

- ~~HW: Limited performance or flexibility.~~
  - ~~ARM-based --- insufficient computing speed.~~
  - ~~ASIC-based --- limited to specific apps.~~
- Runtime: Lack of crucial supports.
  - ~~Protection:~~

    ~~Drive prog. may access unwarranted data.~~
  - Virtualization:

    Support simultaneous multiple drive progs.
- ~~Programming: Lack of a simple abstraction.~~
  - ~~Not integrated with the existing system interface.~~
  - ~~➢Requires considerable code modifications.~~

- HW: FPGA-based.

  12X perf., 31X cost efficiency.

- Runtime:

  A separate control plane that enforces permission check and resource scheduling.

➢ Programming:

A File-based abstraction for in-storage computing.

System Design

Choosing In-Storage Computing Unit

- GPU, ARM, X86, ASIC, FPGA…?

Choosing In-Storage Computing Unit

- GPU, ARM, X86, ASIC, FPGA…?

➢Requirements of the in-storage computing unit.

Choosing In-Storage Computing Unit

- GPU, ARM, X86, ASIC, FPGA…?
- Requirements of the in-storage computing unit.
  - ➢High programmability: supports general workloads.

Choosing In-Storage Computing Unit

- GPU, ARM, X86, ASIC, FPGA…?
- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - ➢Massive parallelism: saturates the high drive internal bandwidth.

Choosing In-Storage Computing Unit

- GPU, ARM, X86, ASIC, FPGA…?
- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - Massive parallelism: saturates high drive internal bandwidth.
  - High energy efficiency: storage drive is originally energy-efficient (5-10W).

Choosing In-Storage Computing Unit

- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - Massive parallelism: saturates high drive internal bandwidth.
  - High energy efficiency: storage drive is originally energy-efficient (5-10W).

		GPU	ARM	X86	ASIC	FPGA
Programmability		Good	Good	Good	No	Good
Parallelism	Data-Level	Good	Poor	Fair	Best	Good
	Pipeline-Level	No	No	No	Best	Good
Energy Efficiency		Fair	Fair	Poor	Best	Good

Choosing In-Storage Computing Unit

- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - Massive parallelism: saturates high drive internal bandwidth.
  - High energy efficiency: storage drive is originally energy-efficient (5-10W).

		GPU	ARM	X86	ASIC	FPGA
Programmability		Good	Good	Good	No	**Good**
Parallelism	Data-Level	Good	Poor	Fair	**Best**	Good
	Pipeline-Level	No	No	No	**Best**	Good
Energy Efficiency		Fair	Fair	Poor	**Best**	Good

Choosing In-Storage Computing Unit

- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - Massive parallelism: saturates high drive internal bandwidth.
  - High energy efficiency: storage drive is originally energy-efficient (5-10W).

		GPU	ARM	X86	ASIC	FPGA
Programmability		Good	Good	Good	No	Good
Parallelism	Data-Level	Good	Poor	Fair	Best	**Good**
	Pipeline-Level	No	No	No	Best	**Good**
Energy Efficiency		Fair	Fair	Poor	Best	**Good**

Choosing In-Storage Computing Unit

- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - Massive parallelism: saturates high drive internal bandwidth.
  - High energy efficiency: storage drive is originally energy-efficient (5-10W).

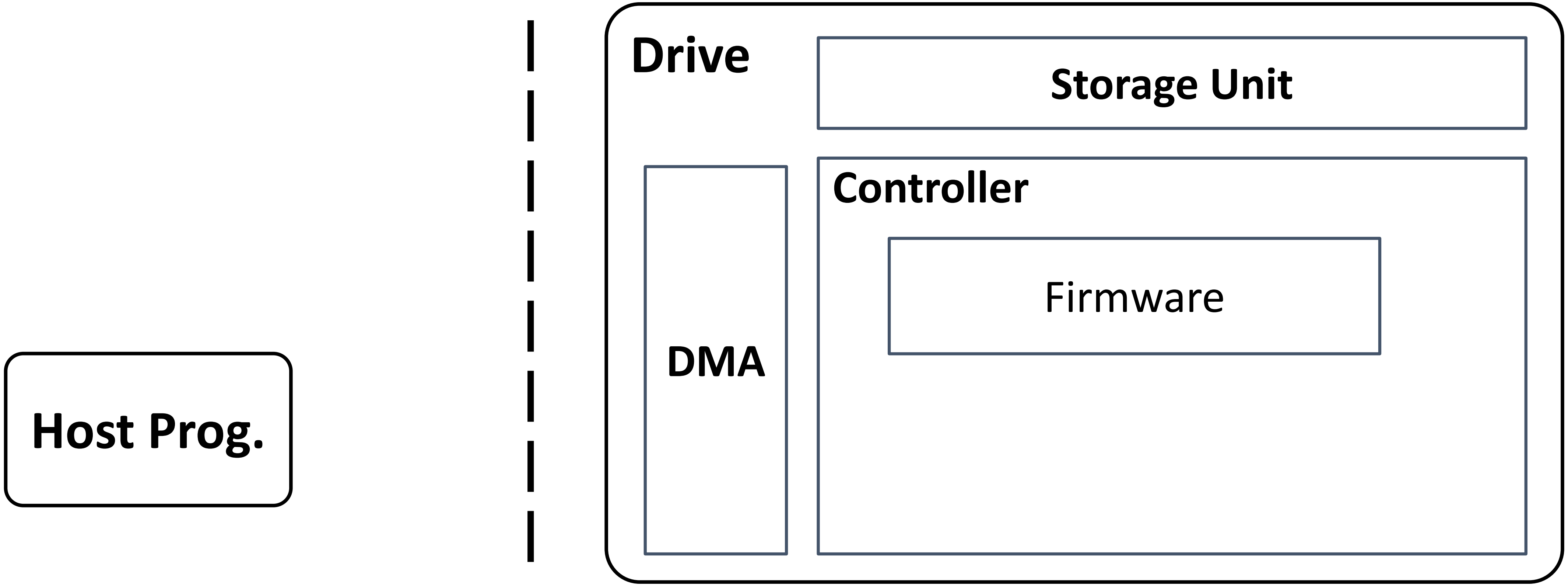
		GPU	ARM	X86	ASIC	FPGA
Programmability		Good	Good	Good	No	Good
Parallelism	Data-Level	Good	Poor	Fair	Best	Good
	Pipeline-Level	No	No	No	Best	**Good**
Energy Efficiency		Fair	Fair	Poor	Best	**Good**

Choosing In-Storage Computing Unit

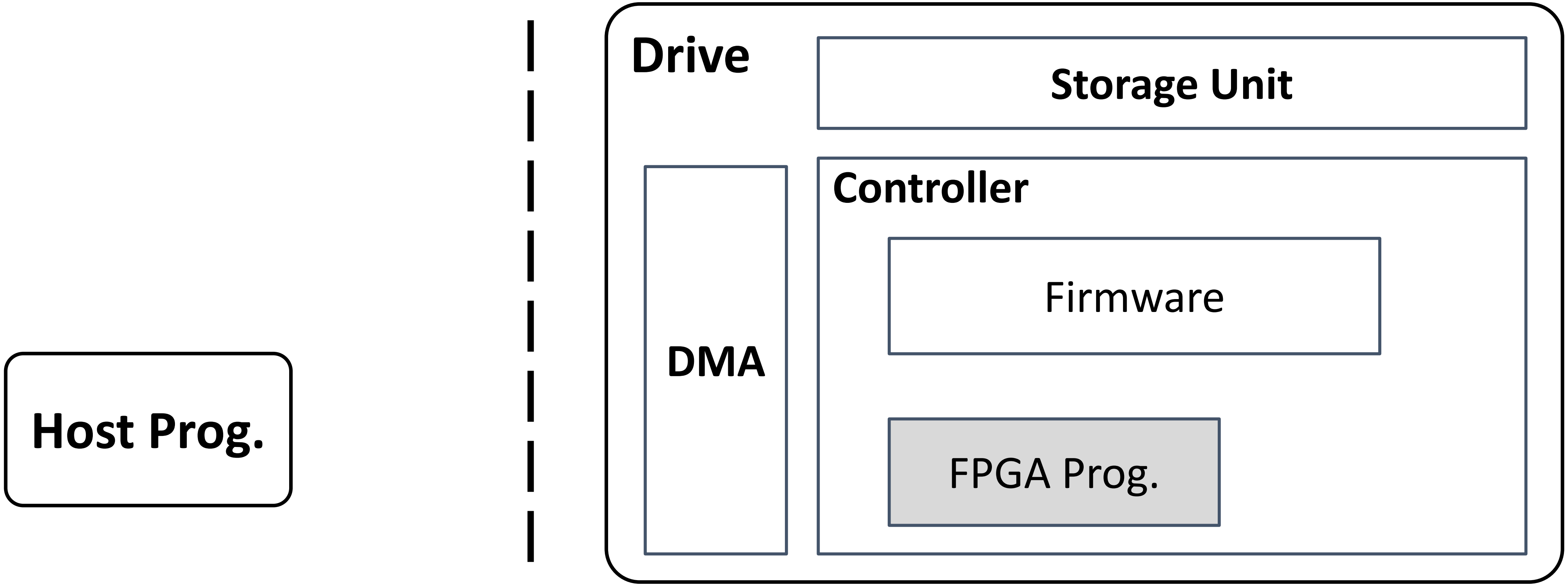
- Requirements of the in-storage computing unit.
  - High programmability: supports general workloads.
  - Massive parallelism: saturates high drive internal bandwidth.
  - High energy efficiency: storage drive is originally energy-efficient (5-10W).

		GPU	ARM	X86	ASIC	FPGA ✓
Programmability		Good	Good	Good	No	Good
Parallelism	Data-Level	Good	Poor	Fair	Best	Good
	Pipeline-Level	No	No	No	Best	Good
Energy Efficiency		Fair	Fair	Poor	Best	Good

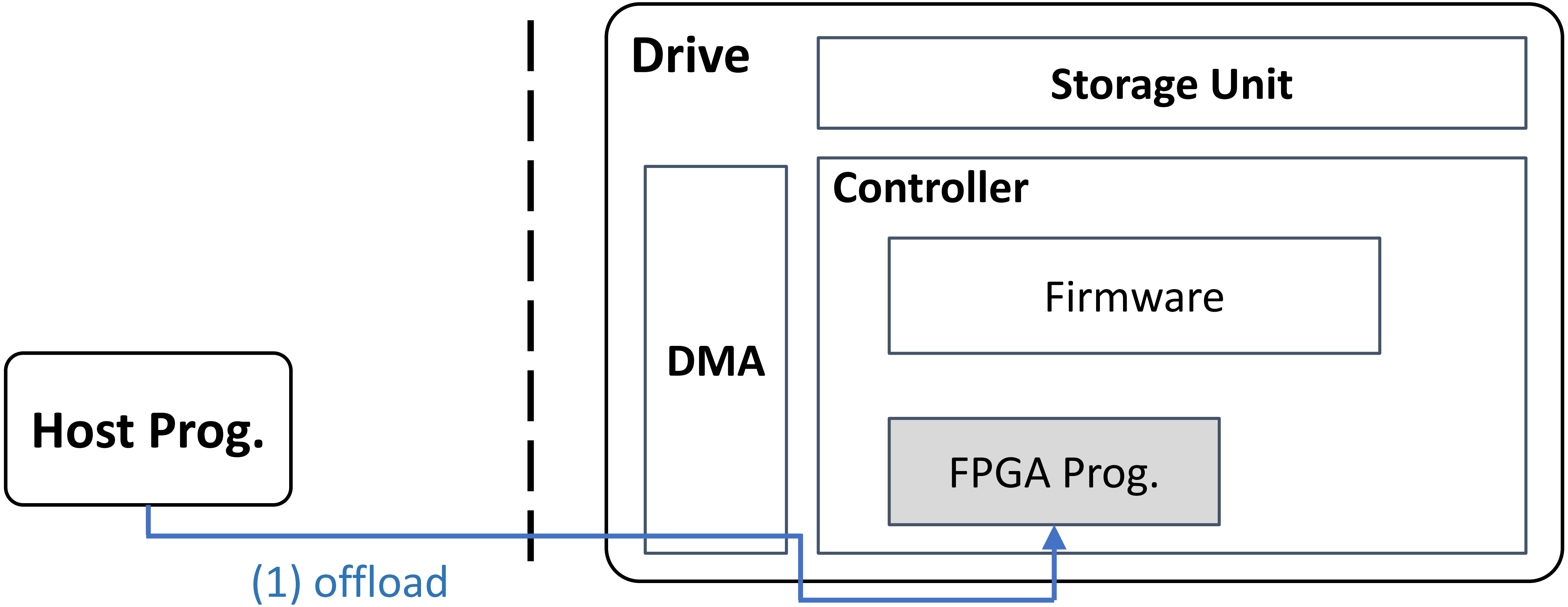
The Initial System Architecture



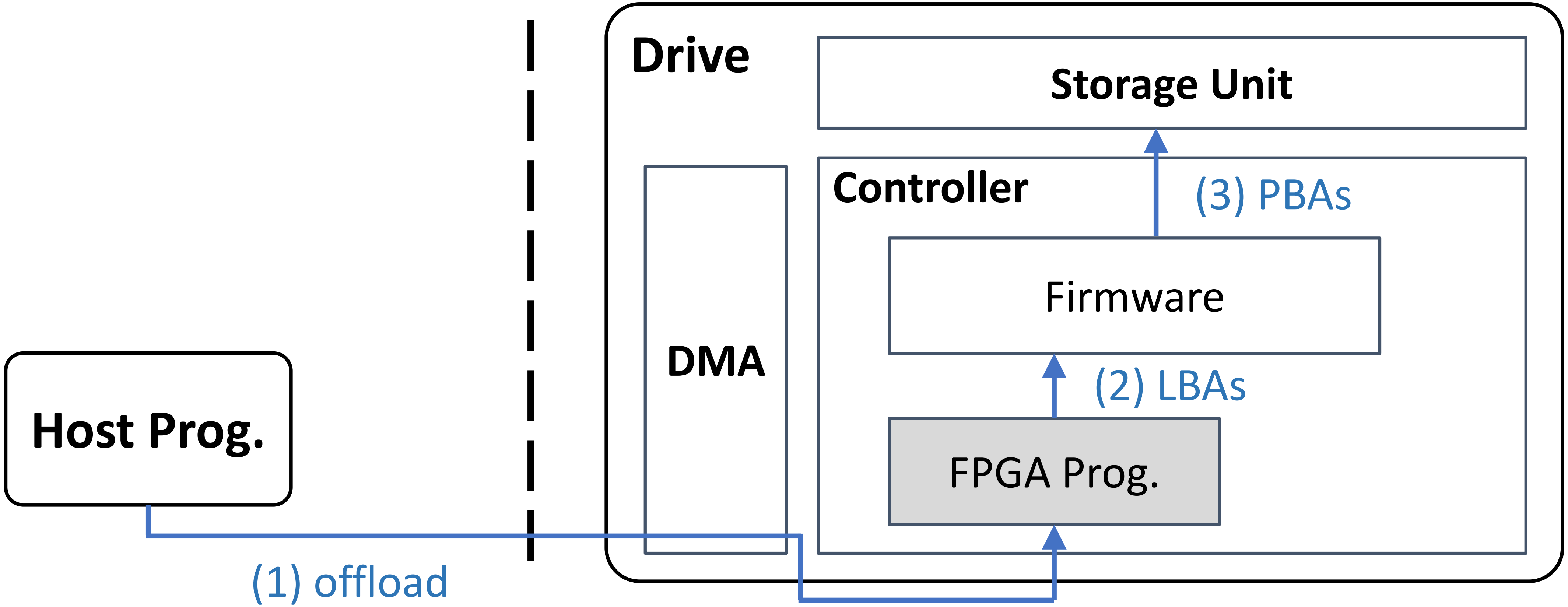
The Initial System Architecture



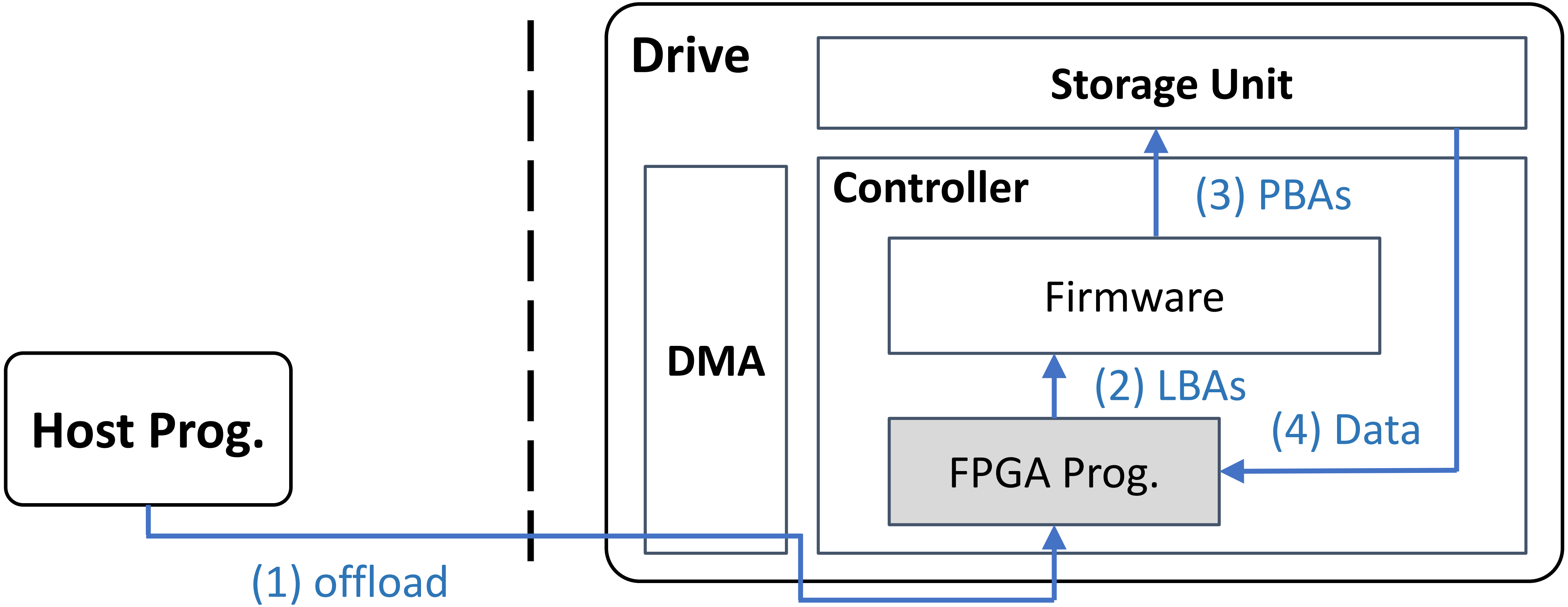
The Initial System Architecture



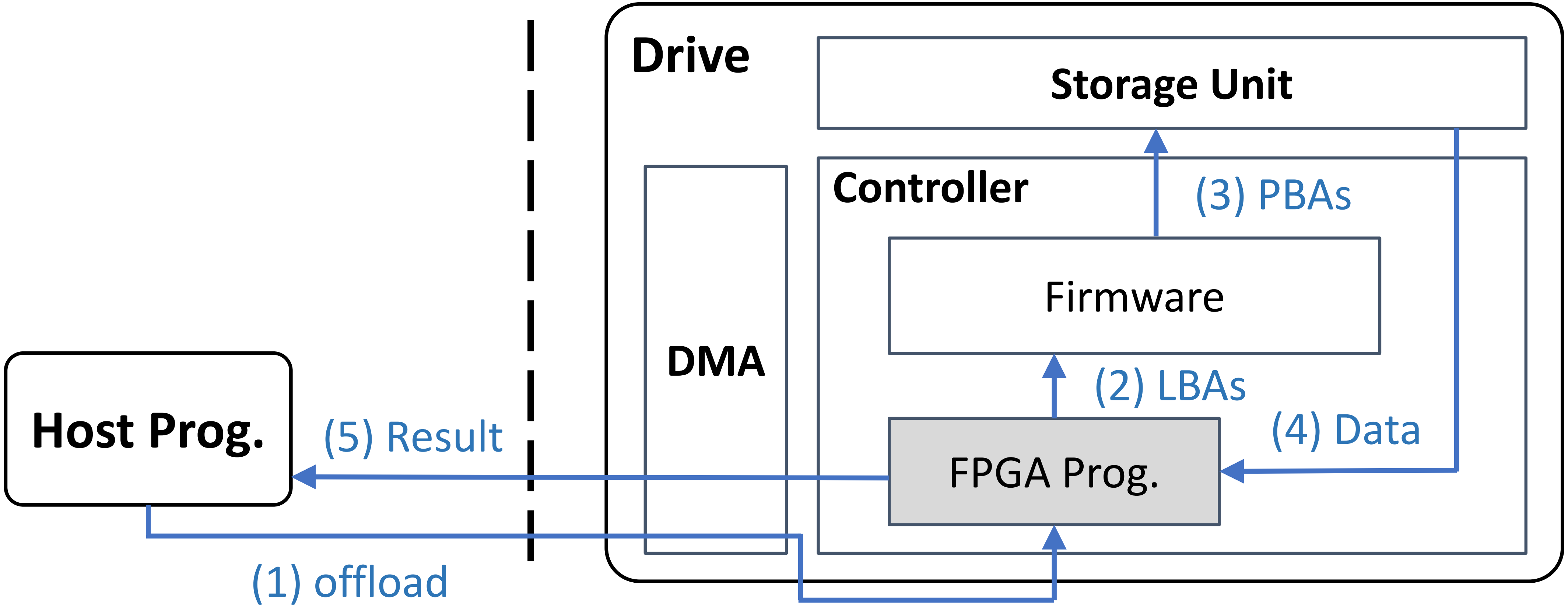
The Initial System Architecture



The Initial System Architecture

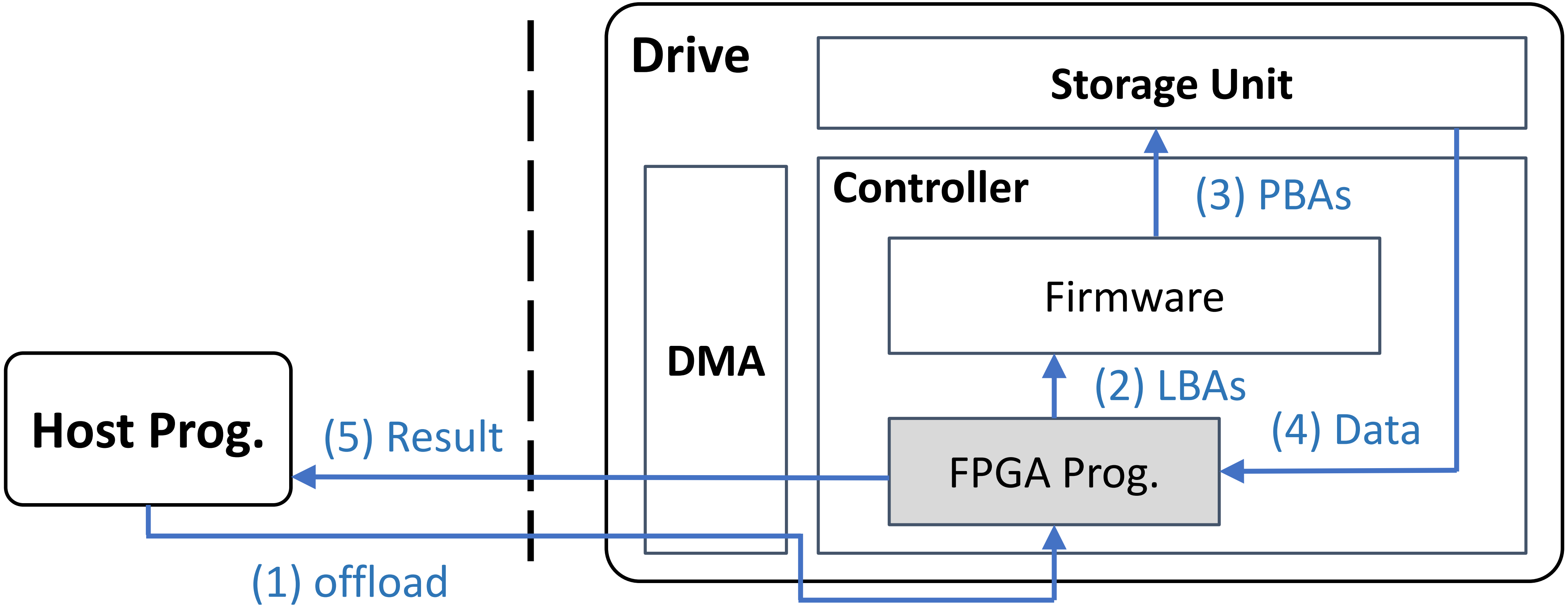


The Initial System Architecture



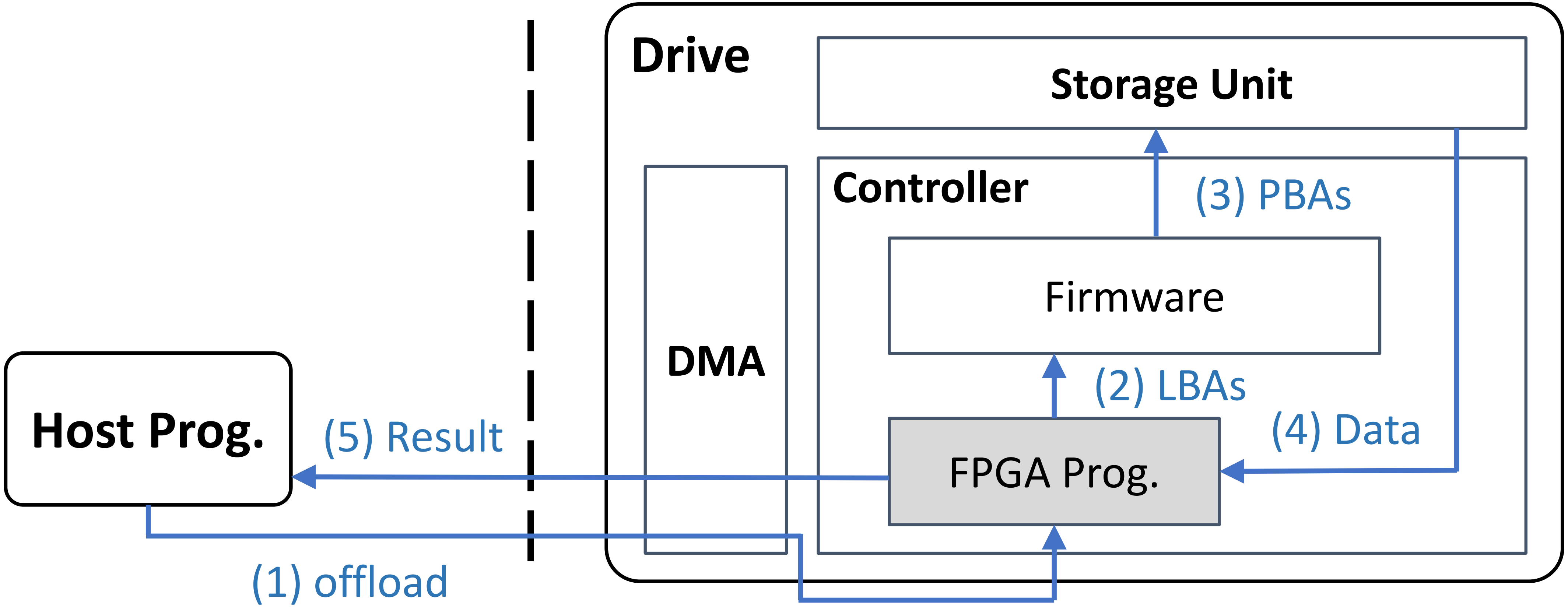
The Initial System Architecture

- Lacks of protection.
  - ➢Drive program can issue arbitrary storage I/O requests.



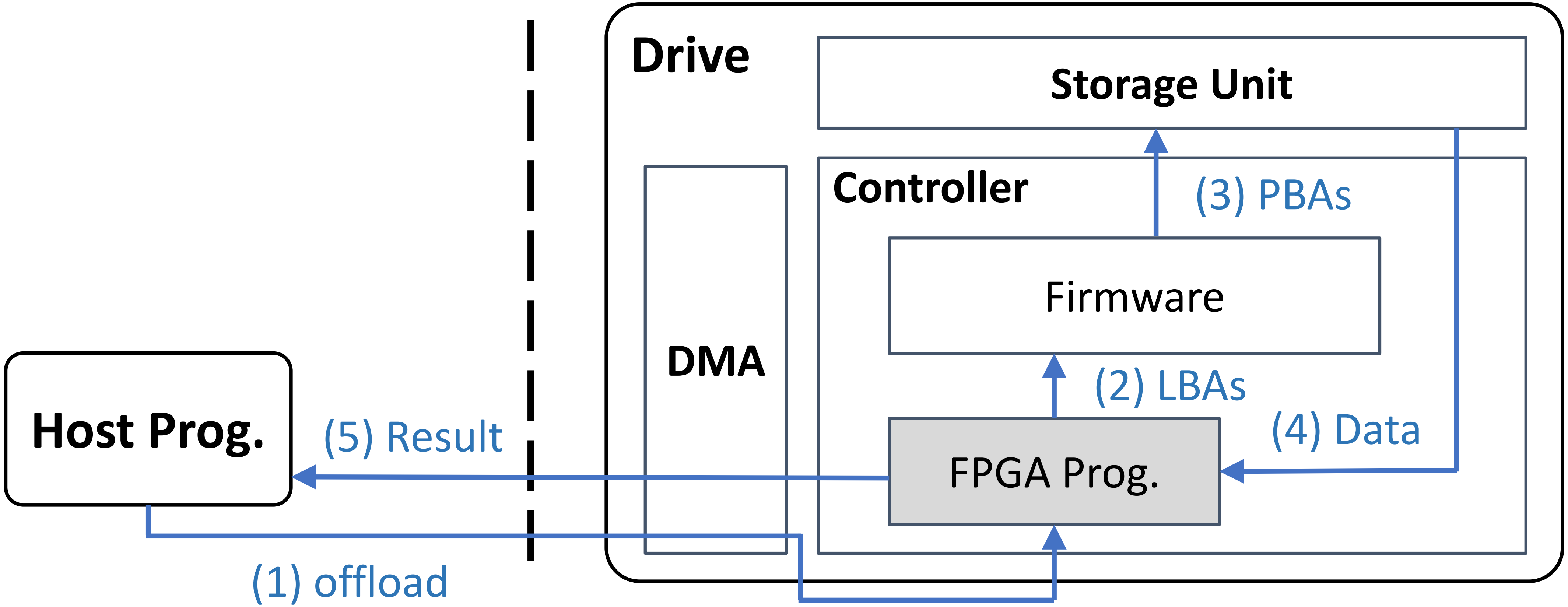
The Initial System Architecture

- Lacks of protection.
  - Drive program can issue arbitrary storage I/O requests.
  - ➢Need a **control plane** to enforce system policies.



Separate *Control Plane* and Data Plane

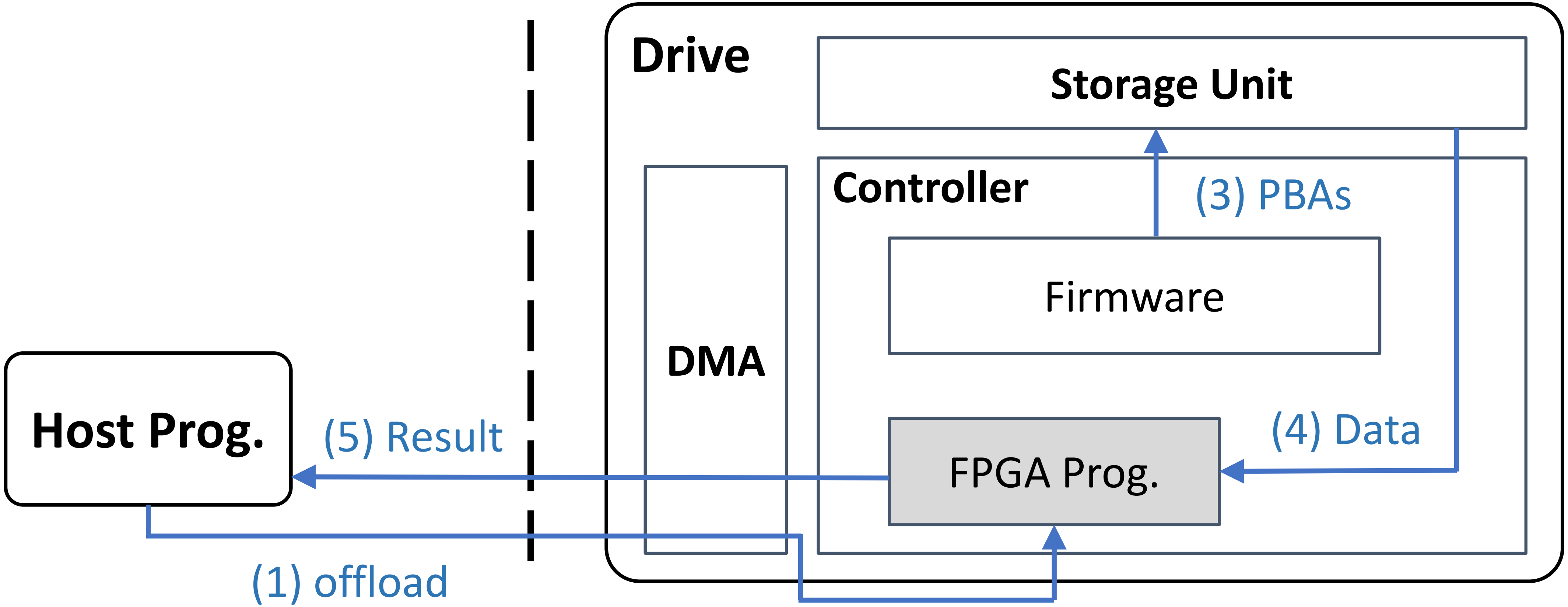
- Make drive program "compute-only".



Separate *Control Plane* and Data Plane

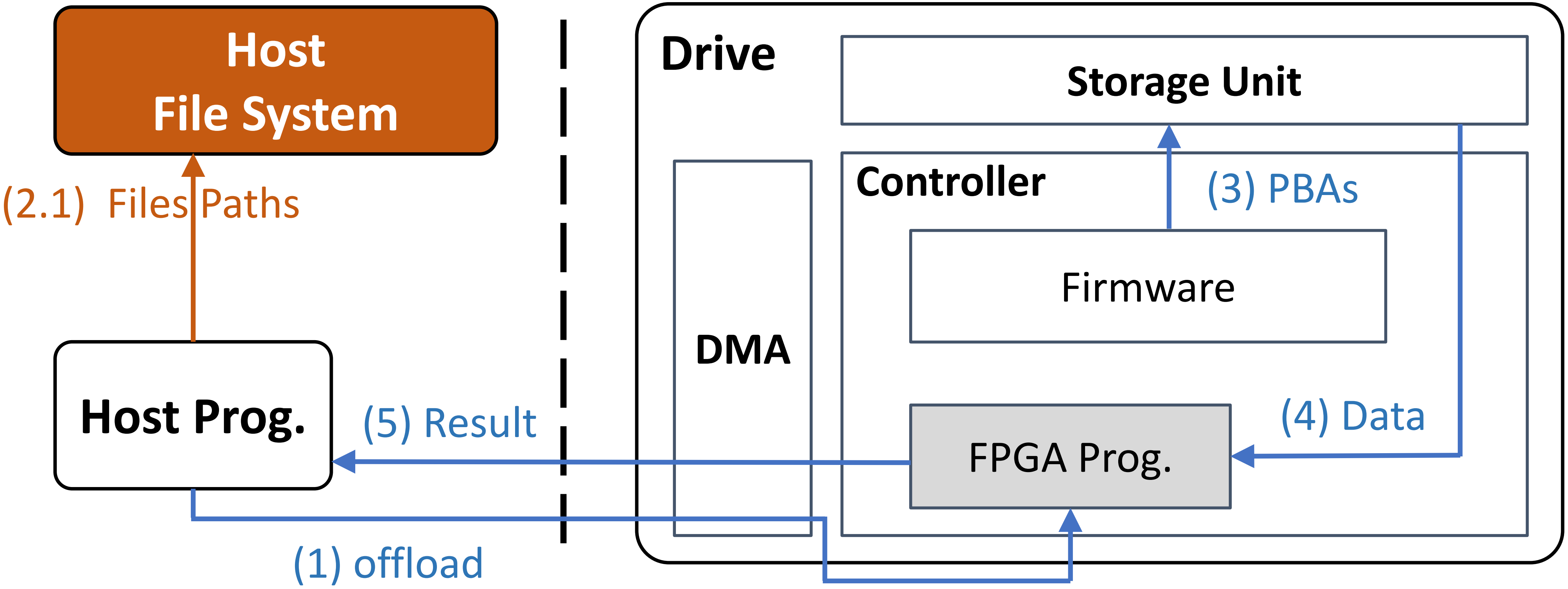
- Make drive program “compute-only”.

➢The control plane is responsible for issuing storage I/O requests.



Separate *Control Plane* and Data Plane

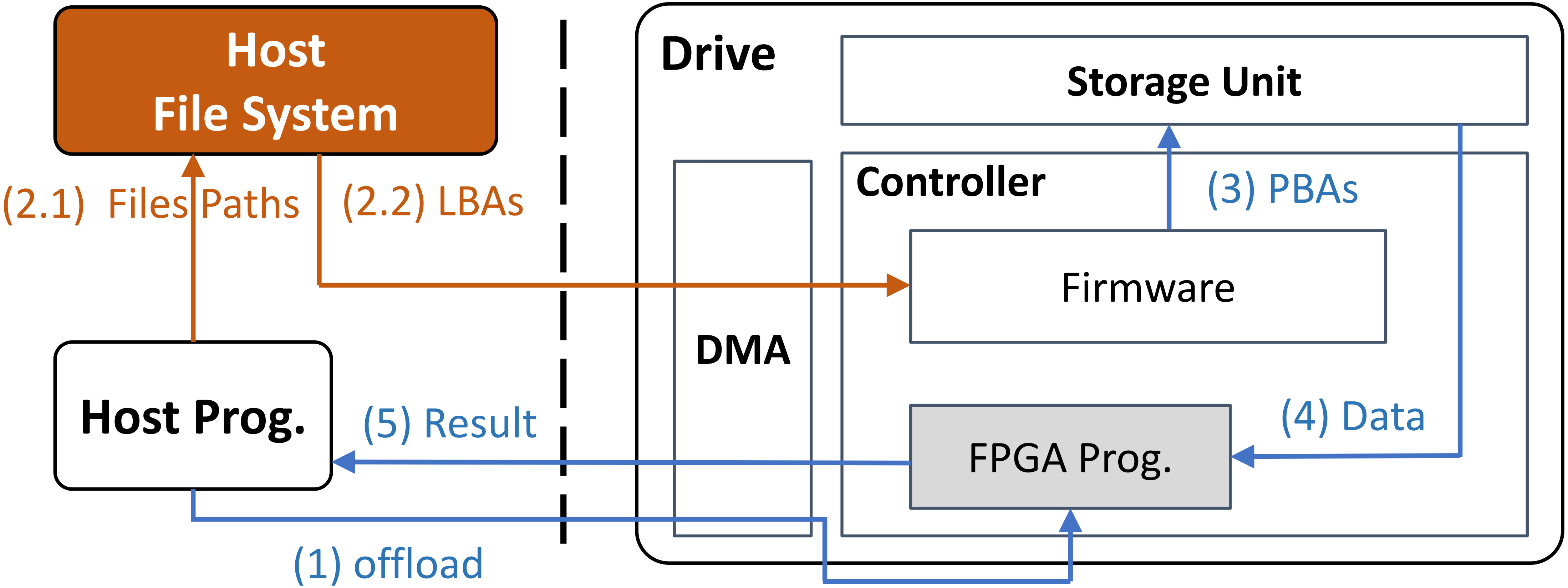
- Host file system performs permission check on requested input files.



Separate *Control Plane* and Data Plane

- Host file system performs permission check on requested input files.

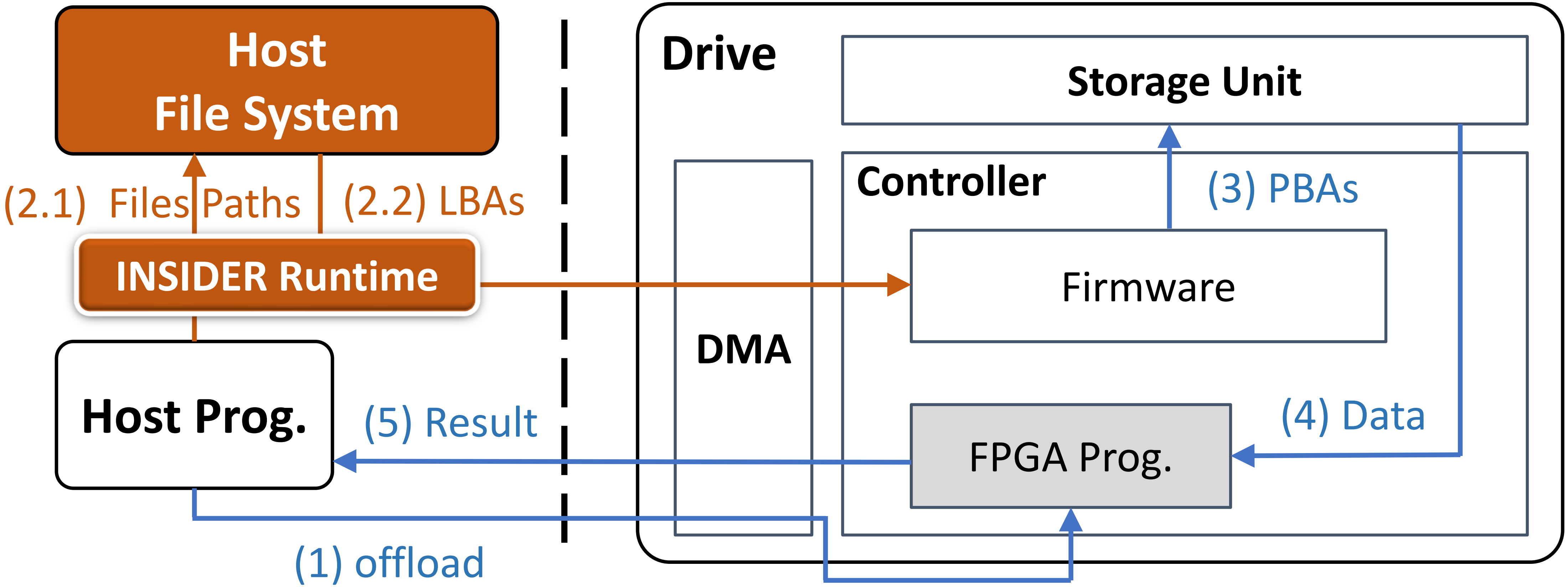
➢ Corresponding LBAs are sent to drive firmware to issue storage I/Os.



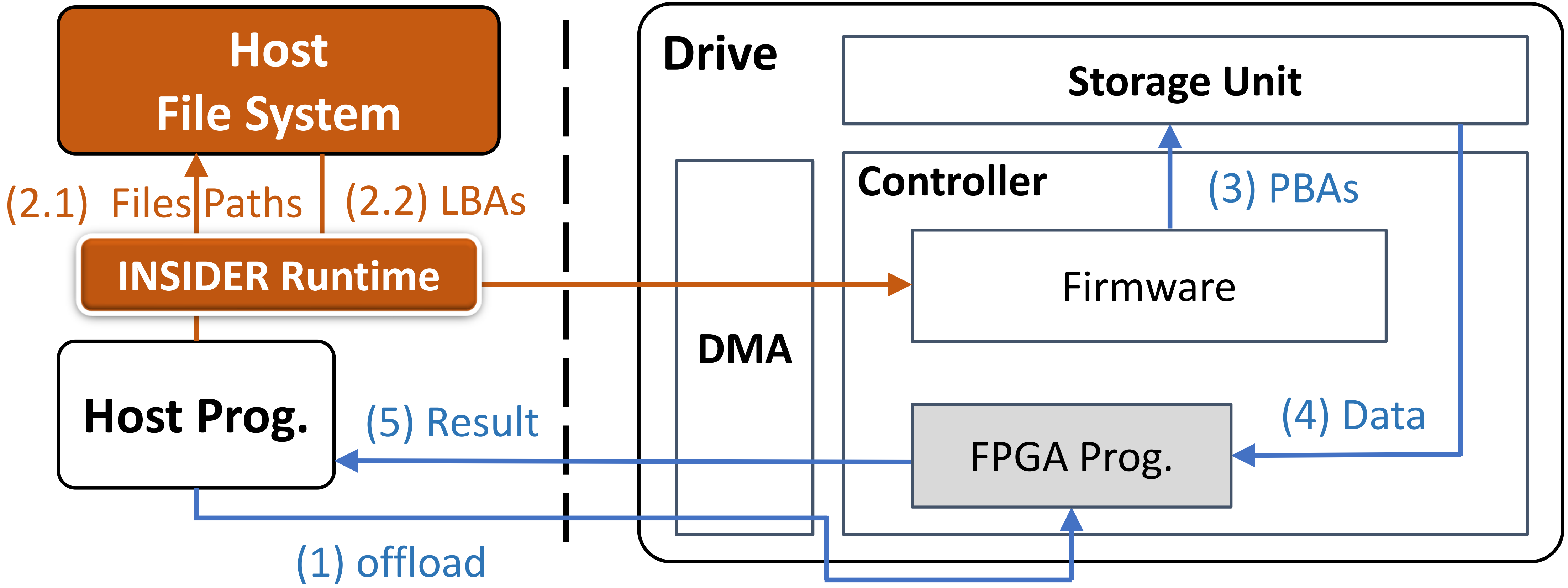
Separate *Control Plane* and Data Plane

- Host file system performs permission check on requested input files.
- Corresponding LBAs are sent to drive firmware to issue storage I/Os.

➢ Enforced by our trusted runtime component.

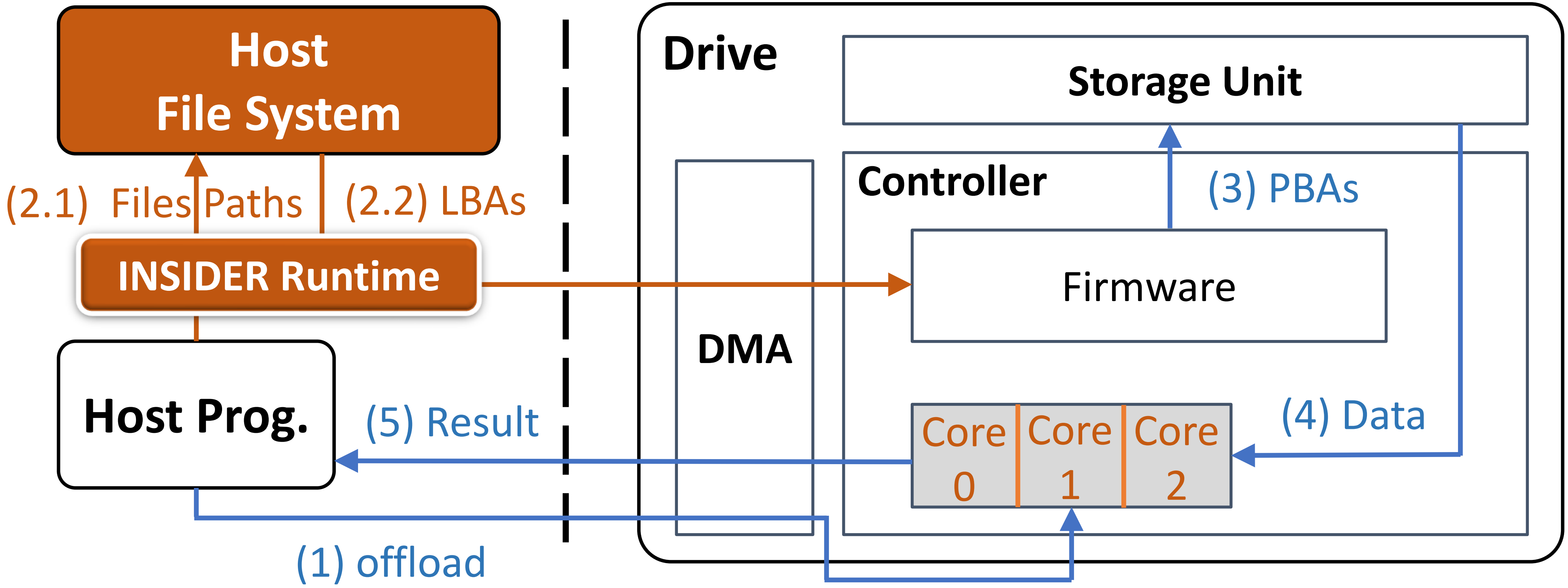


Extend the Control Plane to Support *Virtualization*



Extend the Control Plane to Support *Virtualization*

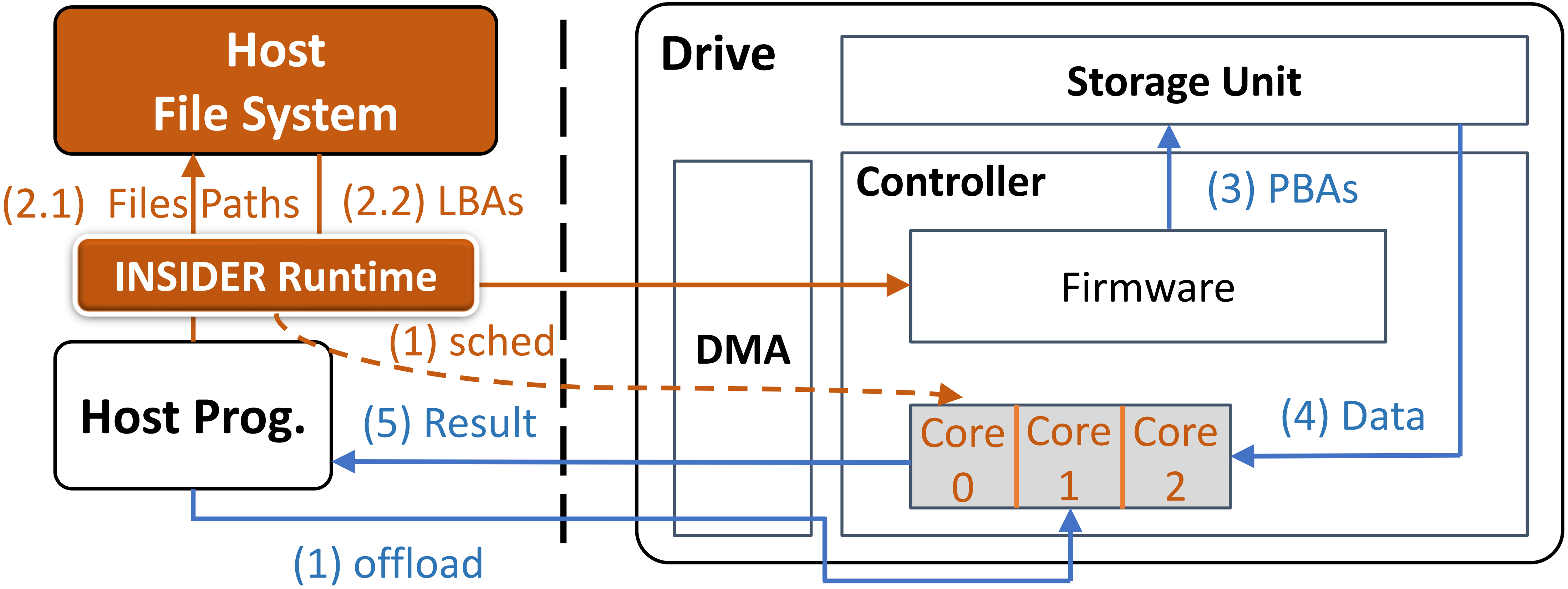
- Leverage *partial reconfiguration* to enable a “multi-core” FPGA.



Extend the Control Plane to Support *Virtualization*

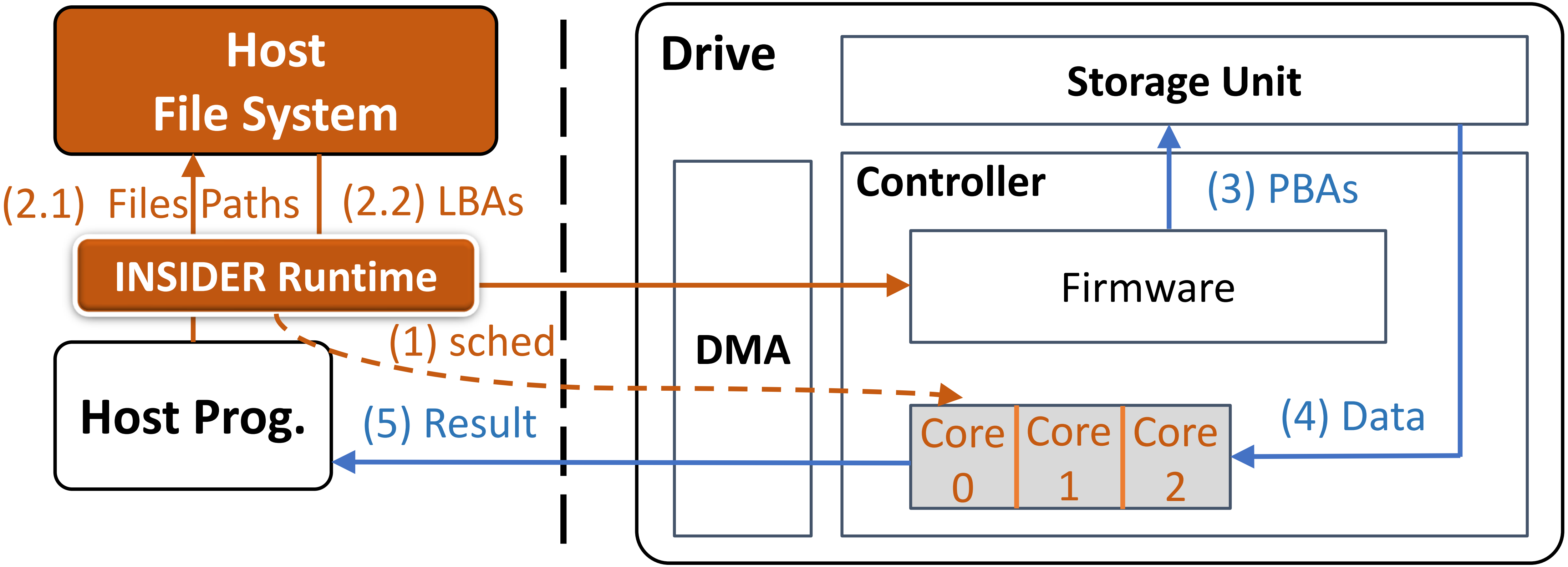
- Leverage *partial reconfiguration* to enable a “multi-core” FPGA.

➢Host runtime enforces drive task scheduling centrally.



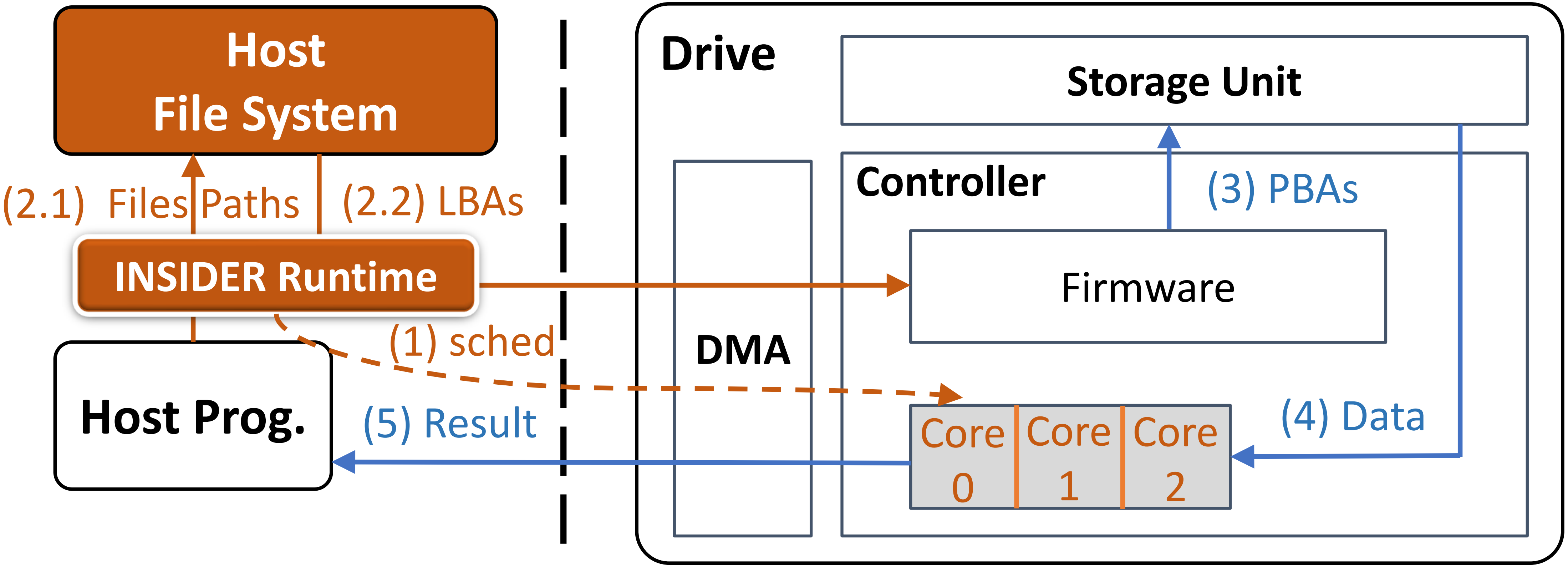
Extend the Control Plane to Support *Virtualization*

- Requires drive bandwidth scheduling among drive processes.



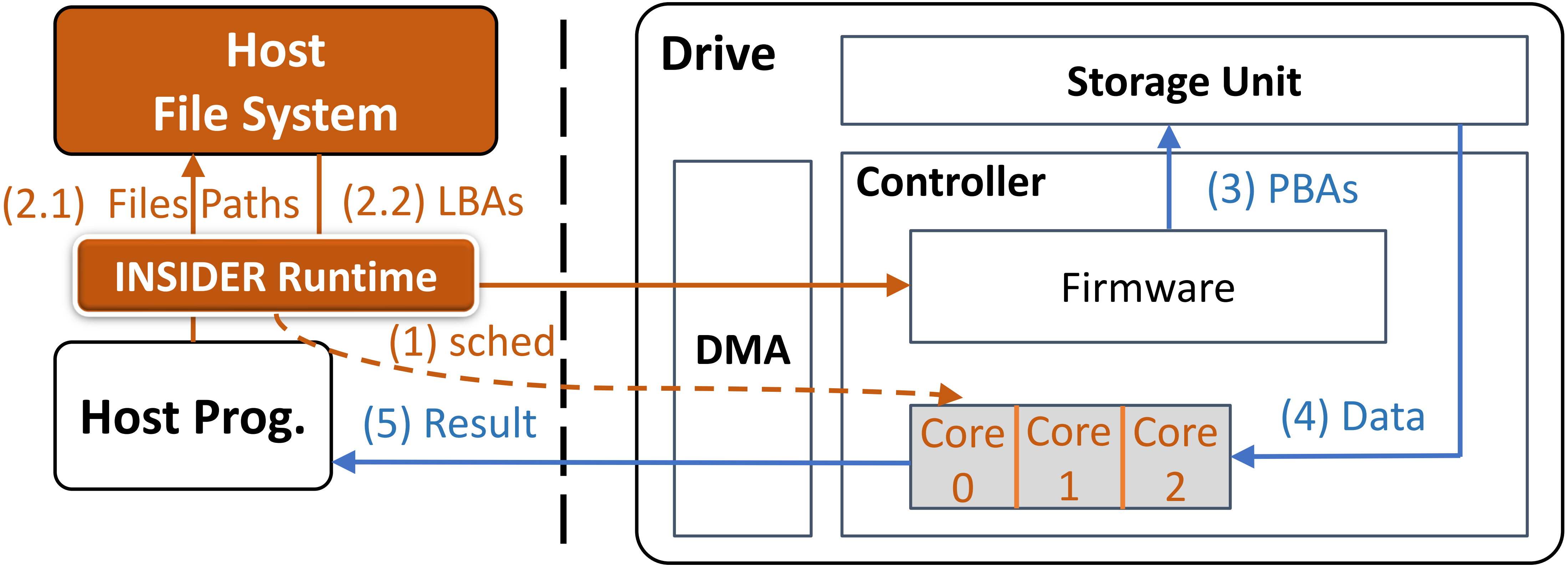
Extend the Control Plane to Support *Virtualization*

- Requires drive bandwidth scheduling among drive processes.
  - Adaptive and fair.



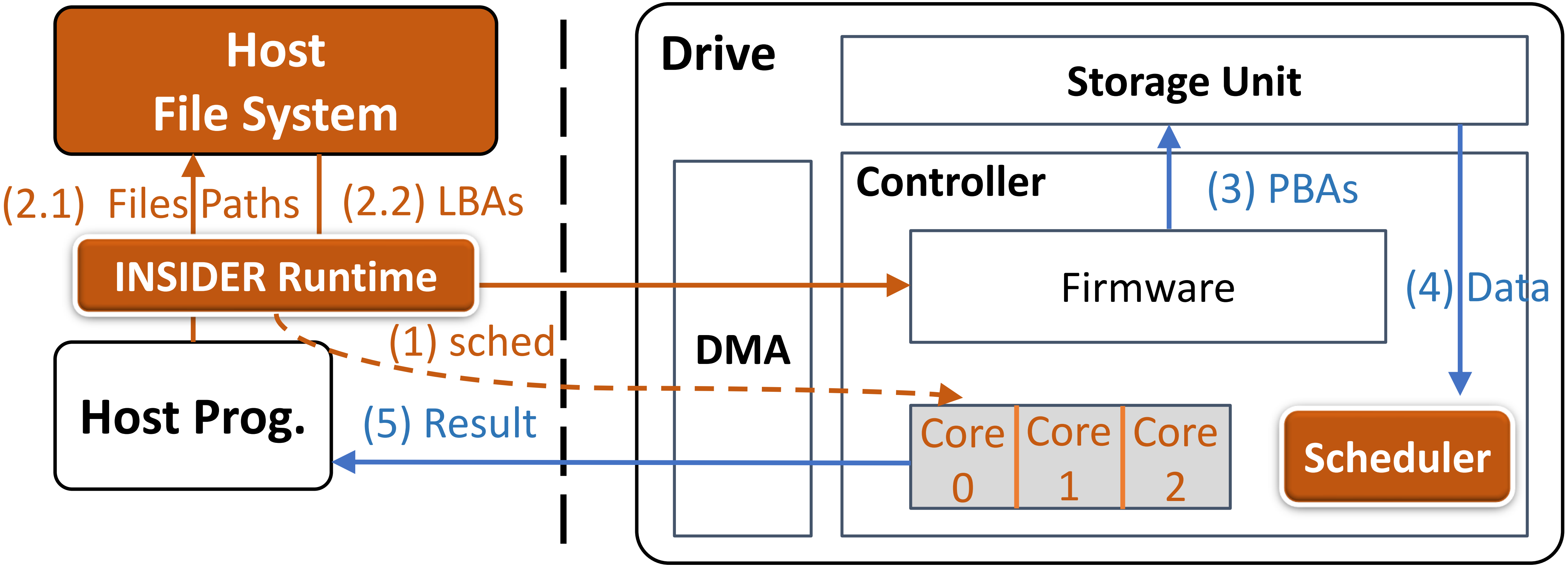
Extend the Control Plane to Support *Virtualization*

- Requires drive bandwidth scheduling among drive processes.
  - Adaptive and fair.
  - ➤Cannot do at host-side INSIDER runtime --- too slow, PCIe RTT is 1 µs.



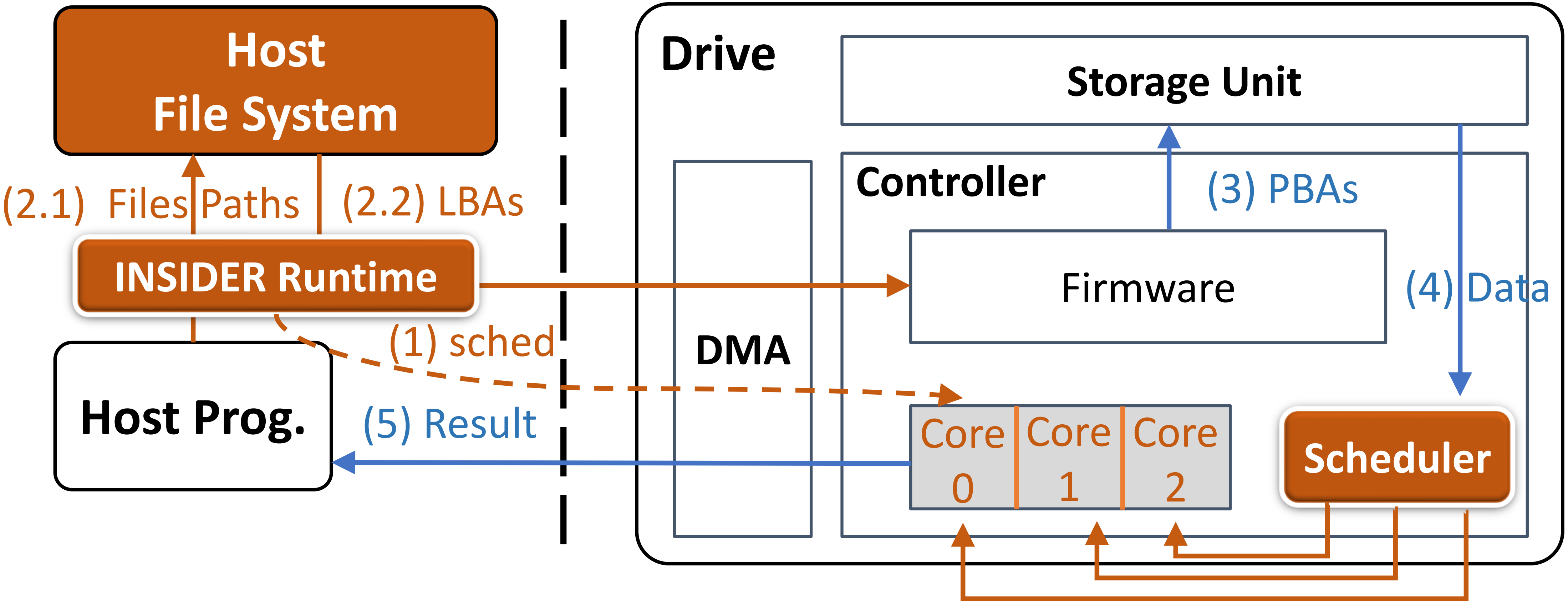
Extend the Control Plane to Support *Virtualization*

- Partially offload control plane into the FPGA hardware.



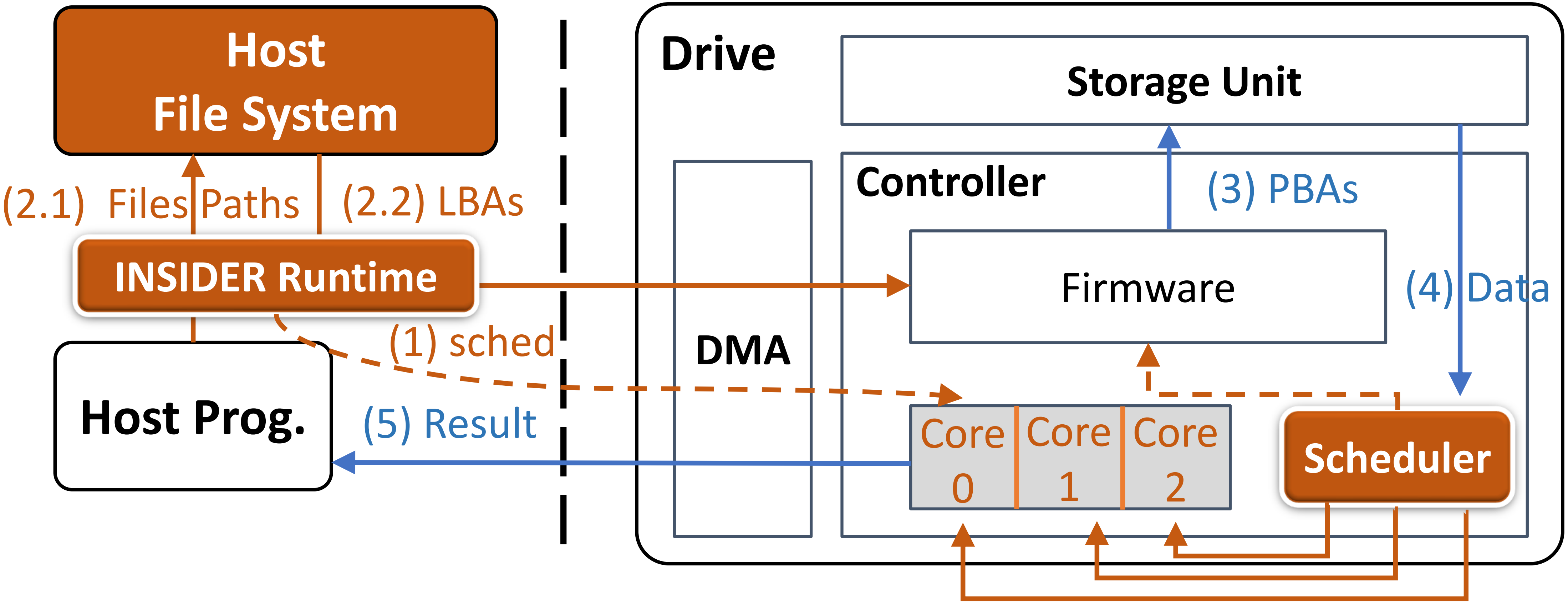
Extend the Control Plane to Support *Virtualization*

- Partially offload control plane into the FPGA hardware.
  - ➢Monitors the drive bw consumption by using the dispatching knowledge.



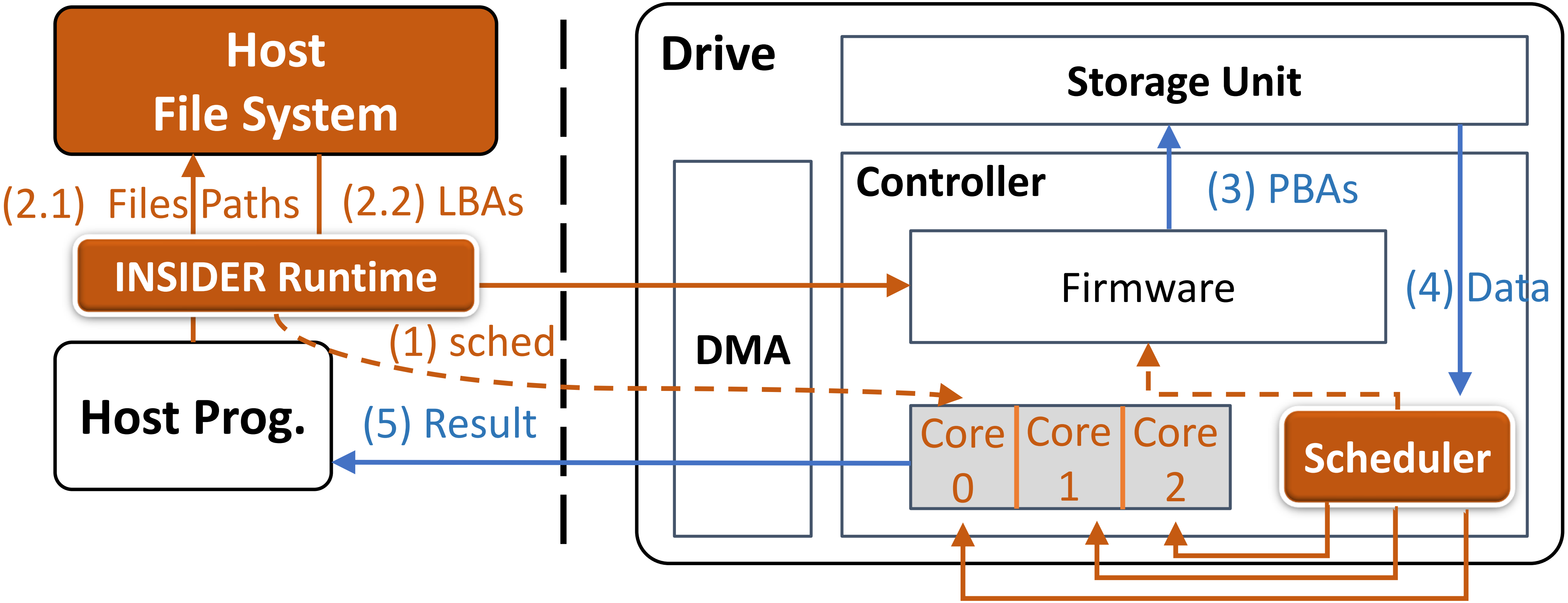
Extend the Control Plane to Support *Virtualization*

- Partially offload control plane into the FPGA hardware.
  - Monitors the drive bw consumption by using the dispatching knowledge.
  - ➢Provides feedback to firmware to adjust the req rate.



Extend the Control Plane to Support *Virtualization*

- Partially offload control plane into the FPGA hardware.
  - Monitors the drive bw consumption by using the dispatching knowledge.
  - Provides feedback to firmware to adjust the req rate.
  - ➢Design a policy similar with deficit round-robin for fairness.



Programming Model --- Virtual File

Programming Model --- Virtual File

- Abstracts in-storage computing as file operations.

Programming Model --- Virtual File

- Abstract as in-storage computing as file operations.
    - ➢virt_file = drive_program(real_file)

Programming Model --- Virtual File

- Abstract as in-storage computing as file operations.
  - virt_file = drive_program(real_file)

➢Host-side POSIX-like APIs:

  - *int **vopen**(const char *path, int flags)*
  - *ssize_t **vread**(int fd, void *buf, size_t count)*
  - *ssize_t **vwrite**(int fd, void *buf, size_t count)*
  - *int **vsync**(int fd)*
  - *int **vclose**(int fd)*
  - *string **reg_virt_file**(string file_path, string acc_id)*

Programming Model --- Virtual File

- Abstract as in-storage computing as file operations.
  - virt_file = drive_program(real_file)

➢ Host-side POSIX-like APIs:

  - *int* ***vopen****(const char *path, int flags)*
  - *ssize_t* ***vread****(int fd, void *buf, size_t count)*
  - *ssize_t* ***vwrite****(int fd, void *buf, size_t count)*
  - *int* ***vsync****(int fd)*
  - *int* ***vclose****(int fd)*
  - *string* ***reg_virt_file****(string file_path, string acc_id)*

➢ Example: feature selection (prune high dim. feature) in ML training.

Programming Model --- Virtual File

- Abstract as in-storage computing as file operations.
  - virt_file = drive_program(real_file)

➢Host-side POSIX-like APIs:

  - *int **vopen**(const char *path, int flags)*
  - *ssize_t **vread**(int fd, void *buf, size_t count)*
  - *ssize_t **vwrite**(int fd, void *buf, size_t count)*
  - *int **vsync**(int fd)*
  - *int **vclose**(int fd)*
  - *string **reg_virt_file**(string file_path, string acc_id)*

- Example: feature selection (prune high dim. feature) in ML training.
  - ➢post_file = *reg_virt_file*(pre_file, acc_feature_selection)

Programming Model --- Virtual File

- Abstract as in-storage computing as file operations.
  - virt_file = drive_program(real_file)

➢Host-side POSIX-like APIs:

  - *int* ***vopen****(const char *path, int flags)*
  - *ssize_t* ***vread****(int fd, void *buf, size_t count)*
  - *ssize_t* ***vwrite****(int fd, void *buf, size_t count)*
  - *int* ***vsync****(int fd)*
  - *int* ***vclose****(int fd)*
  - *string* ***reg_virt_file****(string file_path, string acc_id)*

- Example: feature selection in ML training.
  - post_file = *reg_virt_file*(pre_file, acc_feature_selection)

  ➢*SVM*(post_file)

Evaluation

Experiment Setup

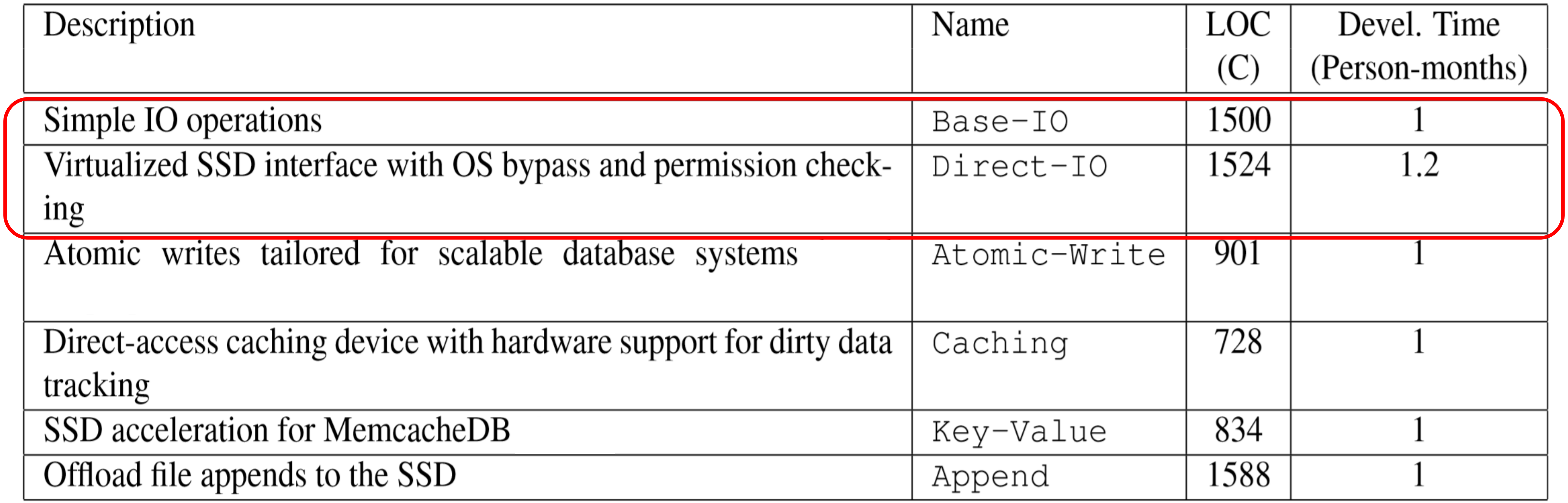
- Build an in-storage computing drive using a PCIe-based FPGA board.

Capacity	64 GB
Latency	5 µs
Sequential R/W	16 GB/s
Host/Drive Bus	PCIe Gen3 **x8** *and* x16
Host File System	XFS

Applications and Their Development Efforts

Application	Devel.Time (Person-Day)	LOC	
		Host	Drive
Grep	3	51	193
KNN	2	77	72
Statistics	3	65	170
SQL Query	5	97	256
Data Integration	5	41	307
Feature Selection	9	50	632
Bitmap file decompression	5	94	213

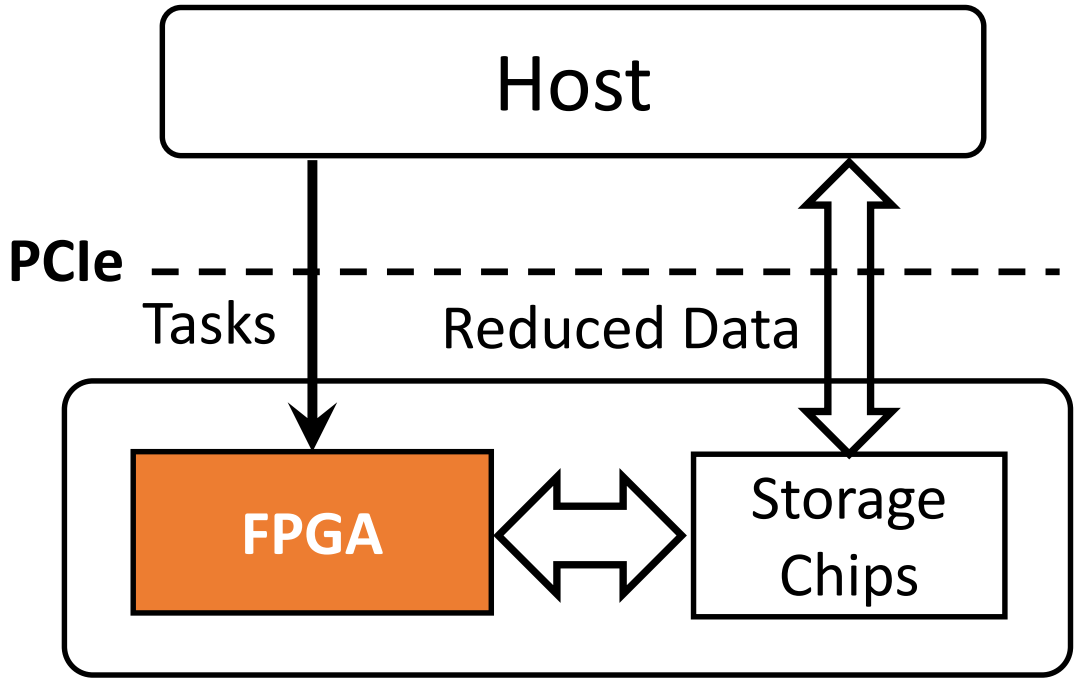
Applications and Their Development Efforts



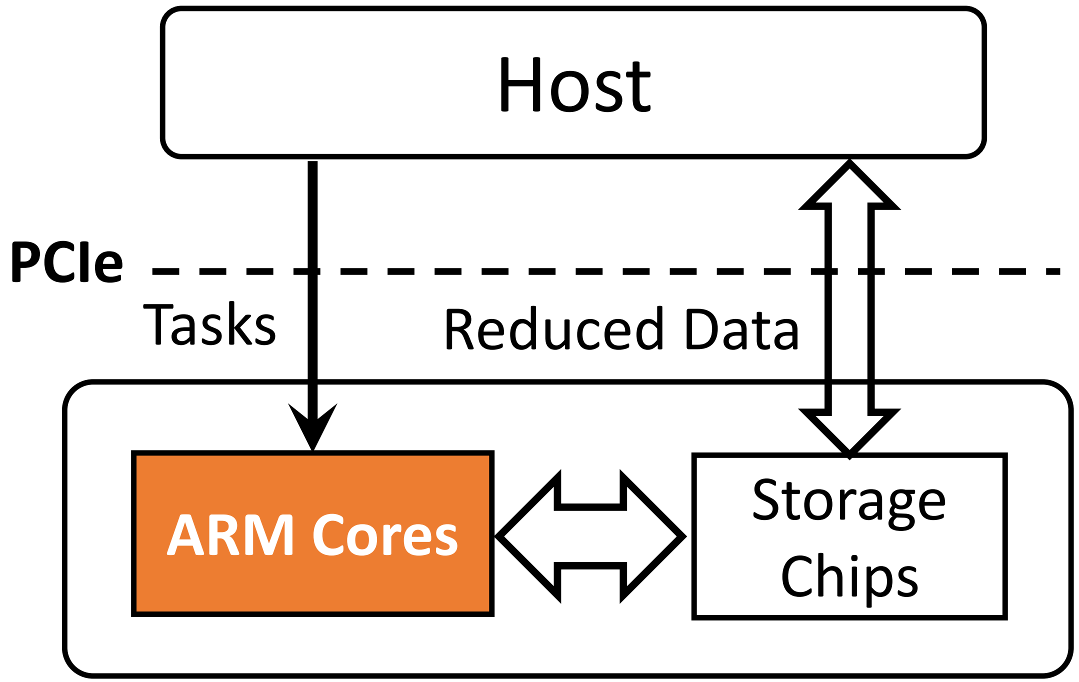
Description	Name	LOC (C)	Devel. Time (Person-months)
Simple IO operations	`Base-IO`	1500	1
Virtualized SSD interface with OS bypass and permission checking	`Direct-IO`	1524	1.2
Atomic writes tailored for scalable database systems	`Atomic-Write`	901	1
Direct-access caching device with hardware support for dirty data tracking	`Caching`	728	1
SSD acceleration for MemcacheDB	`Key-Value`	834	1
Offload file appends to the SSD	`Append`	1588	1

Taken from Willow [OSDI'14].

INSIDER vs ARM-ISC

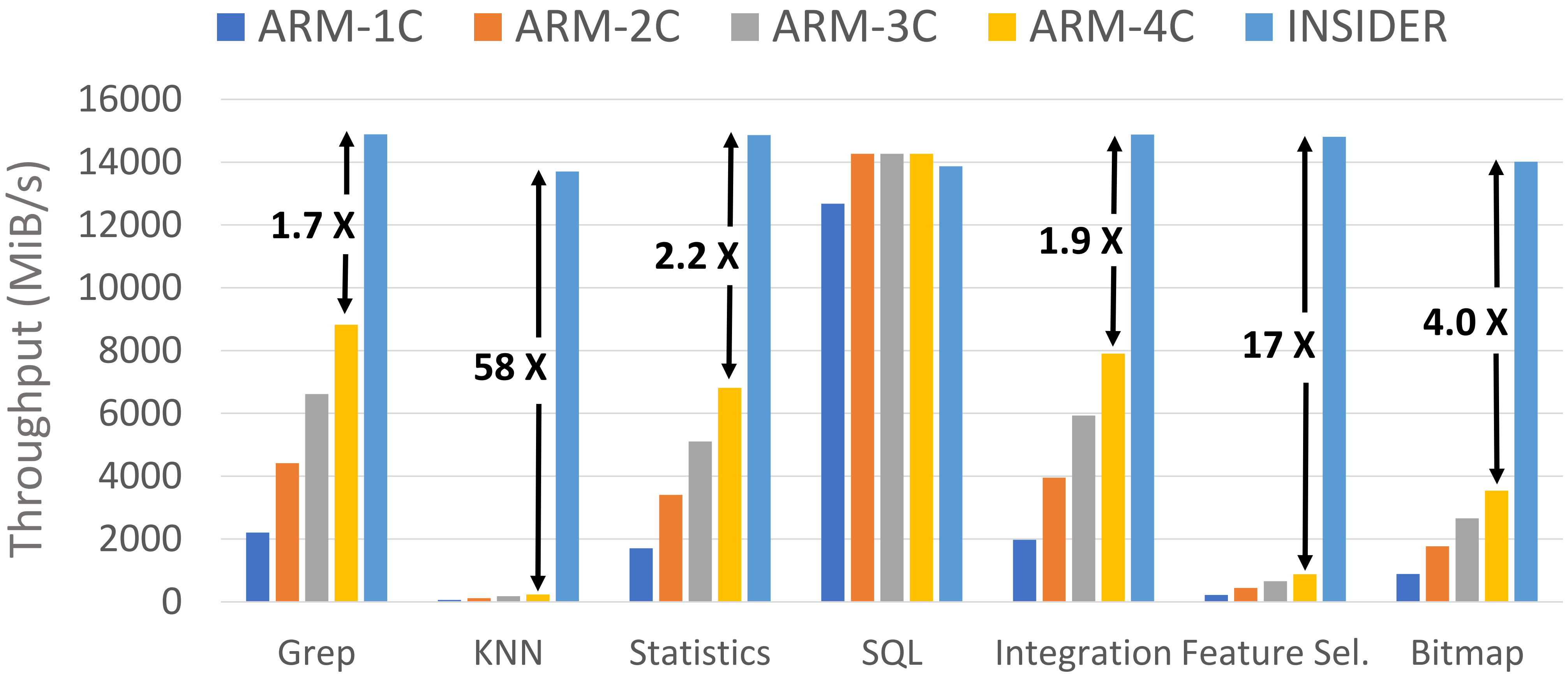


INSIDER (Xilinx Virtex / Artix)



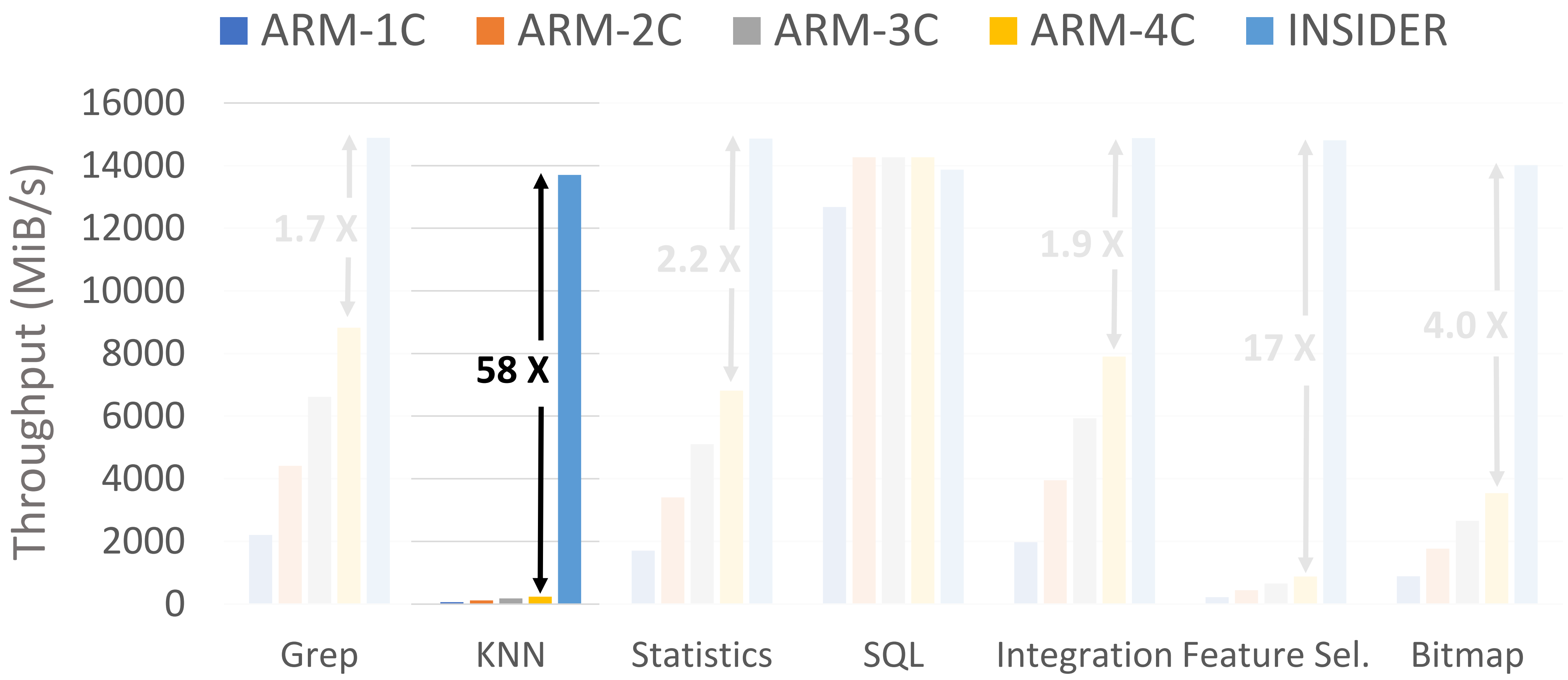
ARM-ISC (Cortex-A72)

Throughput (INSIDER vs ARM-ISC)

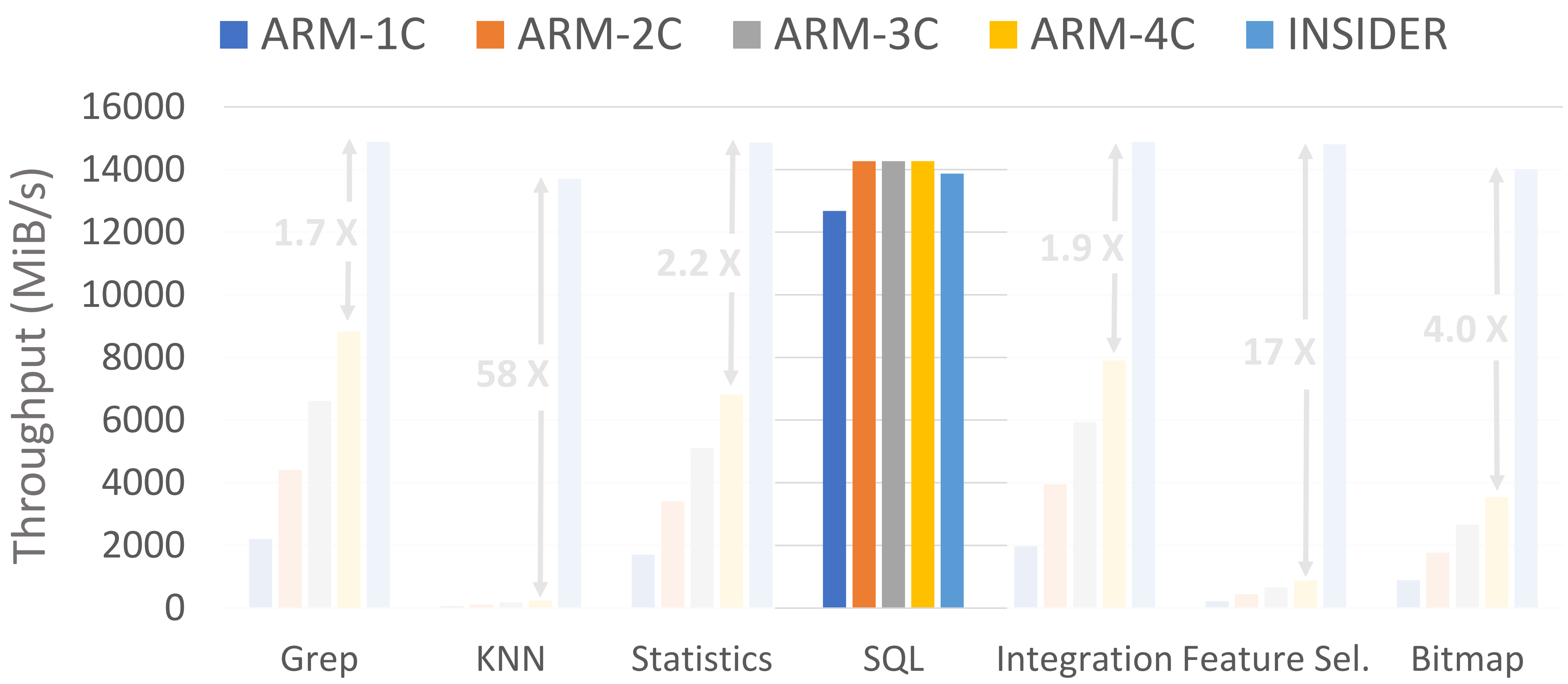


12 X performance on average

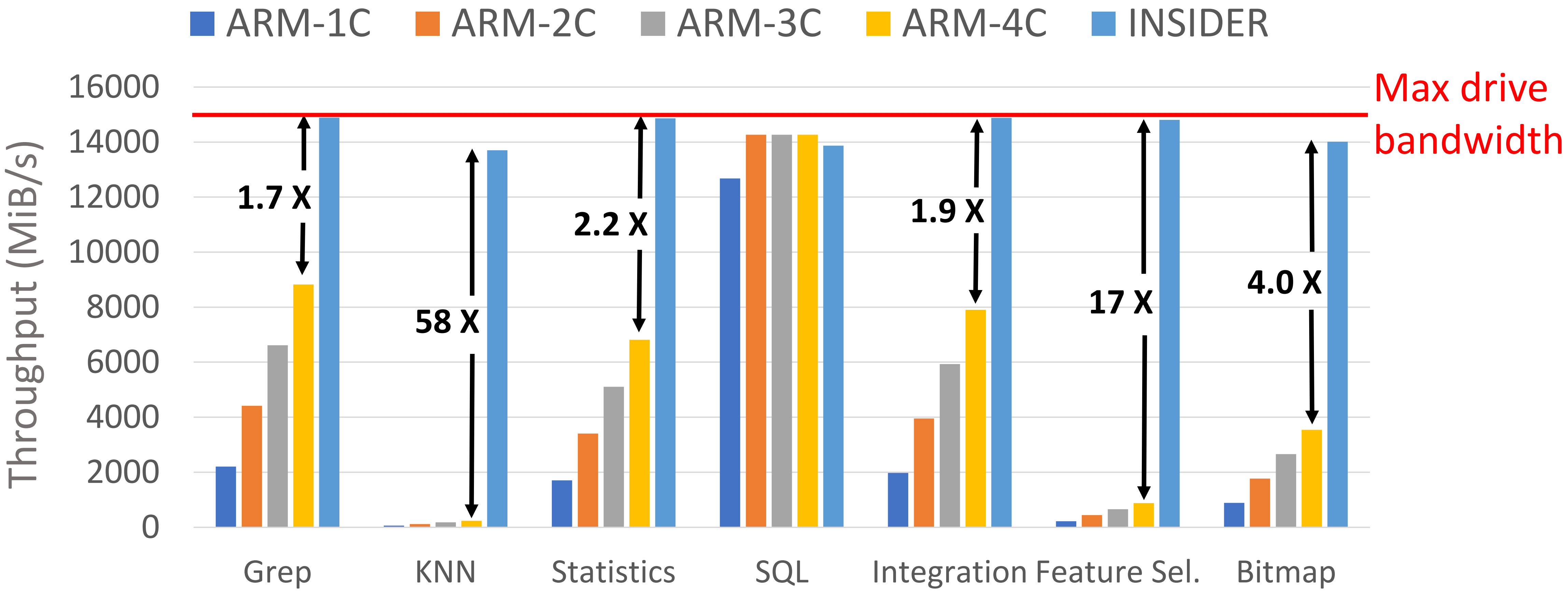
Throughput (INSIDER vs ARM-ISC)



Throughput (INSIDER vs ARM-ISC)



Throughput (INSIDER vs ARM-ISC)



Cost Efficiency (INSIDER vs ARM-ISC)

- Cost efficiency = throughput / dollars

Cost Efficiency (INSIDER vs ARM-ISC)

- Cost efficiency = throughput / dollars

➢ Use the wholesale price in the evaluation.

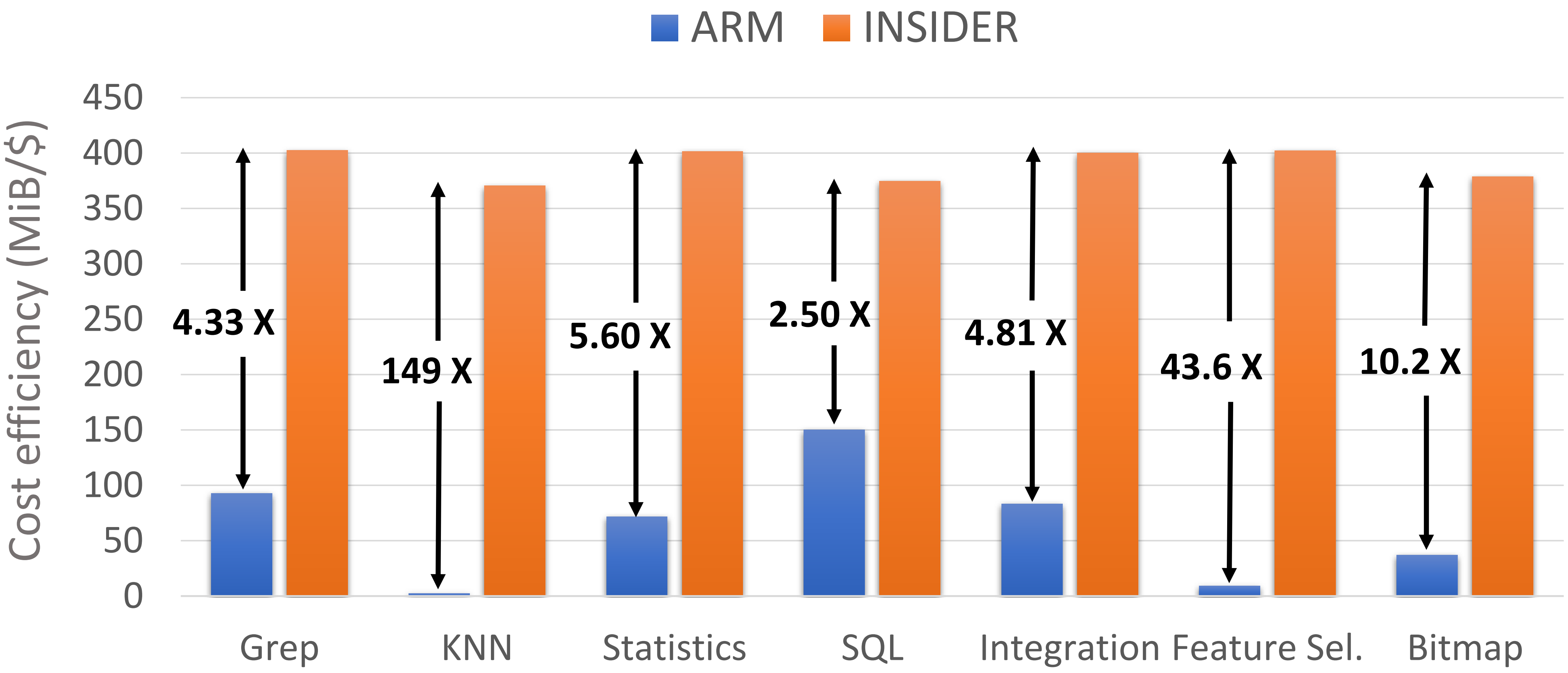
  - Xilinx Artix-7 XC7A200T: $37.

  (https://www.alibaba.com/product-detail/XC7A200T-1FFG1156C-IC-Embedded-FPGA-Field_60730073325.html)

  - ARM Cortex A72 (4 cores, 1.8 GHz): $95.

  (https://www.mouser.com/ProductDetail/NXP-Freescale/LS1046ASN8T1A?qs=sGAEpiMZZMup8ZLti7BNCxtNz7%252BF43hzZlkvLaqOJ8c%3D)

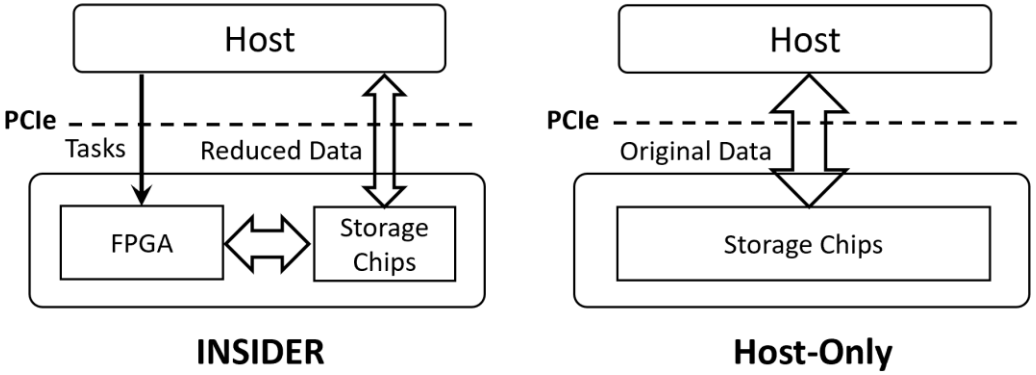
Cost Efficiency (INSIDER vs ARM-ISC)



31 X cost efficiency on average

More Details

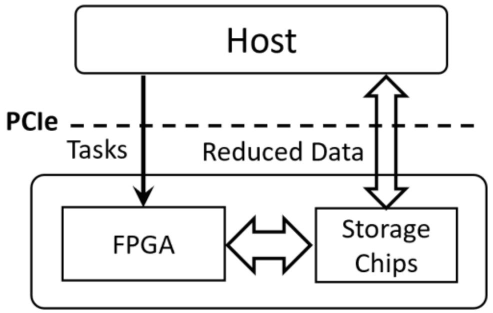
- INSIDER vs the original host-only architecture.



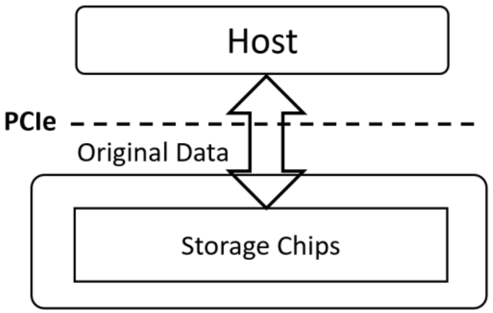
More Details

- INSIDER vs original host-only architecture.

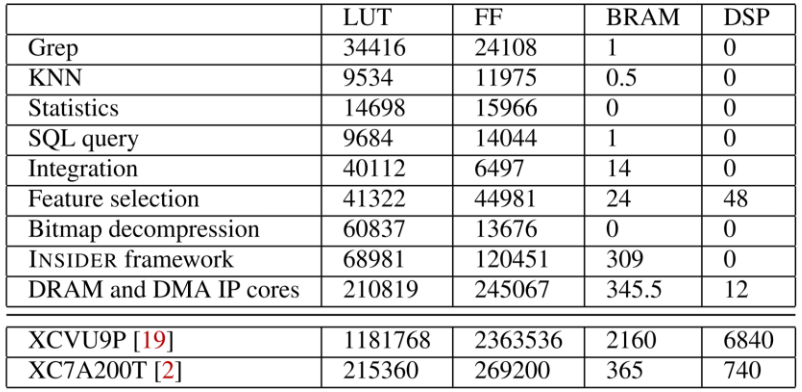
➢ Analysis of FPGA resource utilization.



INSIDER



Host-Only

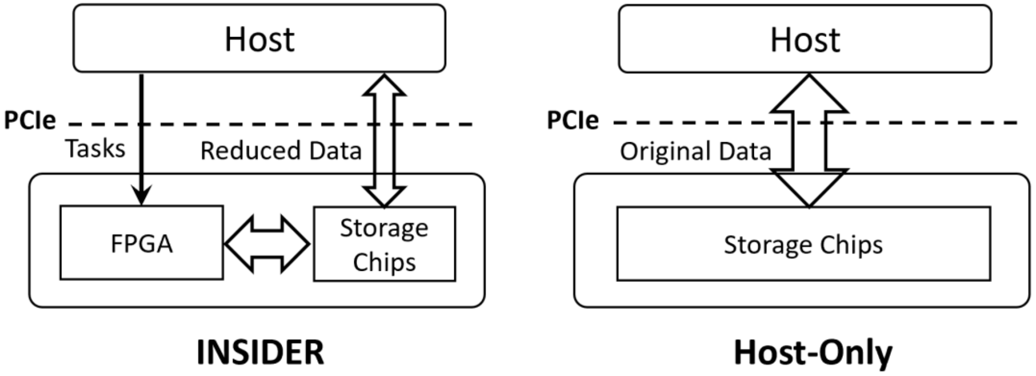


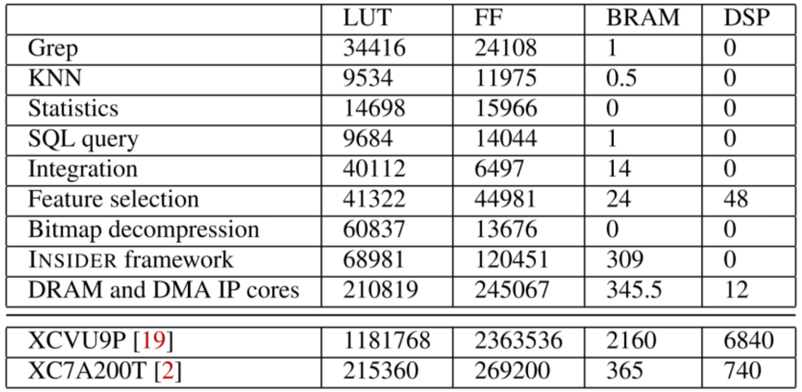
	LUT	FF	BRAM	DSP
Grep	34416	24108	1	0
KNN	9534	11975	0.5	0
Statistics	14698	15966	0	0
SQL query	9684	14044	1	0
Integration	40112	6497	14	0
Feature selection	41322	44981	24	48
Bitmap decompression	60837	13676	0	0
INSIDER framework	68981	120451	309	0
DRAM and DMA IP cores	210819	245067	345.5	12
XCVU9P [19]	1181768	2363536	2160	6840
XC7A200T [2]	215360	269200	365	740

More Details

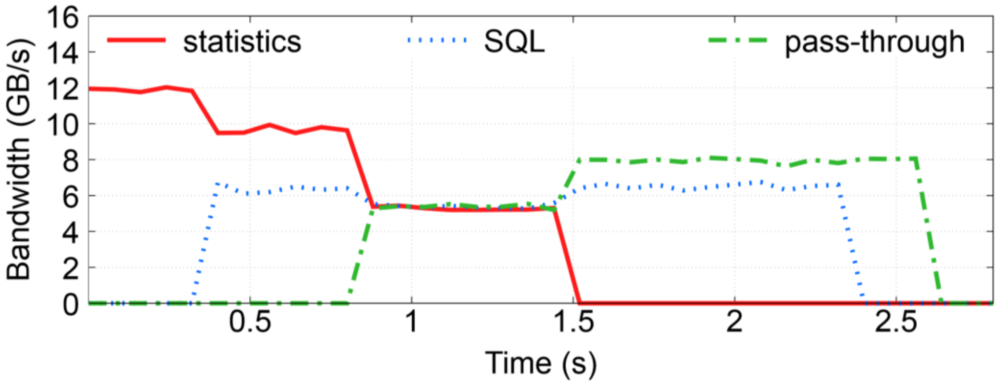
- INSIDER vs original host-only architecture.
- Analysis of FPGA resource utilization.

➢Evaluation of INSIDER’s drive bandwidth scheduler.





	LUT	FF	BRAM	DSP
Grep	34416	24108	1	0
KNN	9534	11975	0.5	0
Statistics	14698	15966	0	0
SQL query	9684	14044	1	0
Integration	40112	6497	14	0
Feature selection	41322	44981	24	48
Bitmap decompression	60837	13676	0	0
INSIDER framework	68981	120451	309	0
DRAM and DMA IP cores	210819	245067	345.5	12
XCVU9P [19]	1181768	2363536	2160	6840
XC7A200T [2]	215360	269200	365	740



...

Conclusion

- “Data movement wall” prevents end users from utilizing the advance in storage technology.

Conclusion

- “Data movement wall” prevents end users from utilizing the advance in storage technology.

➢ We present INSIDER, a full-stack redesigned storage system.

Conclusion

- “Data movement wall” prevents end users from utilizing the advance in storage technology.
- We present INSIDER, a full-stack redesigned storage system.
  - ➢High end-to-end performance and cost efficiency.

Conclusion

- “Data movement wall” prevents end users from utilizing the advance in storage technology.
- We present INSIDER, a full-stack redesigned storage system.
  - High end-to-end performance and cost efficiency.
  - ➢A simple but effective file abstraction for in-storage computing.

Conclusion

- “Data movement wall” prevents end users from utilizing the advance in storage technology.
- We present INSIDER, a full-stack redesigned storage system.
    - High end-to-end performance and cost efficiency.
    - A simple but effective file abstractions for in-storage computing.
    - ➢Enables protection and virtualization for a shared environment.