

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ADVANCED MICRO DEVICES, INC. AND PENSANDO SYSTEMS, INC.,

Petitioners,

v.

CONCURRENT VENTURES, LLC AND XTREAMEDGE, INC.,

Patent Owners.

IPR2025-00223

U.S. Patent No. 10,985,943

PETITION FOR *INTER PARTES* REVIEW

OF U.S. PATENT NO. 10,985,943

TABLE OF CONTENTS

EXHIBIT LIST	4
I. BACKGROUND	6
II. The '943 patent	10
A. Prosecution History	11
B. Priority	12
C. PO's Allegations.....	12
III. Field and Level of Skill	14
IV. Claim Construction.....	15
A. "programmable logic device"	15
V. Invalidity	15
A. Grounds 1-2	16
1. U.S. Patent 10,776,142 ("Seshadri").....	17
2. U.S. Patent Application Publication No. 2018/0191642A1 ("Biederman") 18	
3. Claim 1	20
4. Claim 10	31
5. Claim 2	38
6. Claim 11	40
7. Claim 3	41
8. Claim 12	42
9. Claims 4/13	42
10. Claim 5	45
11. Claim 14	46
12. Claims 9/18	46
B. Grounds 3-4	51
1. U.S. Patent Application Publication 2019/0095113A1 ("Atsatt-113") ..	51
2. Claim 1	52
3. Claim 10	67
4. Claim 2	72

5. Claim 11	74
6. Claims 3/12	74
7. Claims 4/13	75
8. Claims 5/14	76
9. Claims 9/18	77
C. Ground 5	79
1. “Enabling Flexible Network FPGA Clusters in a Heterogeneous Cloud Data Center” (“Tarafdar”).....	79
2. Claim 6	80
3. Claim 15	85
D. Ground 6	86
1. U.S. Patent No. 8,015,386 (“Kulkarni”)	86
2. Claim 7	87
3. Claim 16	91
E. Ground 7	92
1. U.S. Patent Application Publication No. 2019/0041923A1 (“Atsatt-923”) (Ex-1009).....	92
2. Claim 8	93
3. Claim 17	96
VI. Standing	97
VII. Discretionary Analysis	97
VIII. MANDATORY NOTICES AND FEES	98

EXHIBIT LIST

Exhibit No.	Description
1001	U.S. Patent No. 10,985,943 (“’943 patent”)
1002	Prosecution History of the ’943 patent
1003	Declaration of Donald Alpert, Ph.D.
1004	U.S. Patent No. 10,776,142 (“Seshadri”)
1005	U.S. Patent App. Pub. No. 2018/0191642A1 (“Biederman”)
1006	U.S. Patent App. Pub. No. 2019/0095113A1 (“Atsatt-113”)
1007	Tarafdar et al., “Enabling Flexible Network FPGA Clusters in a Heterogeneous Cloud Data Center” (“Tarafdar”)
1008	U.S. Patent No. 8,015,386 (“Kulkarni”)
1009	U.S. Patent App. Pub. No. 2019/0041923A1 (“Atsatt-923”)
1010	Kuon et al., “FPGA Architecture: Survey and Challenges” (“Kuon”)
1011	Trimberger, “Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology” (“Trimberger”)
1012	Teubner & Woods, “Data Processing on FPGAs” (“Teubner”)
1013	Kean, “Secure Configuration of Field Programmable Gate Arrays” (“Kean”)
1014	Frey, “The IBM Data Center Family” (“Frey”)
1015	Desai et al., “BladeCenter system overview” (“Desai”)
1016	Hughes et al., “BladeCenter processor blades, I/O expansion adapters, and units” (“Hughes”)
1017	Caldeira et al., “IBM Power Systems S814 and S824 Technical Overview and Introduction” (Caldeira)
1018	Steffen, “Nallatech Software Tools on the Maxwell FPGA Parallel Computer” (“Steffen”)
1019	Nallatech, “H100 Series FPGA Application Accelerators” (“Nallatech”)
1020	Bassiri & Shahhoseini, “Configuration Reusing in On-Line Task Scheduling for Reconfigurable Computing Systems” (“Bassiri”)
1021	Exhibit D to Concurrent Ventures’ and XstreamEdge’s Oct. 11, 2024 Preliminary Invalidity Contentions
1022	Declaration of S. Delman re Tarafdar
1023	Chen et al., “Enabling FPGAs in the Cloud” (“Chen”)
1024	Asiatici et al., “Virtualized Execution Runtime for FPGA Accelerators in the Cloud” (“Asiatici”)

Exhibit No.	Description
1025	Vipin & Fahmy, “FPGA Dynamic and Partial Reconfiguration: A Survey of Architectures, Methods, and Applications” (“Vipin”)
1026	Fahmy et al., “Virtualized FPGA Accelerators for Efficient Cloud Computing” (“Fahmy”)
1027	RocketBoards, “PCIe Root Port with MSI”
1028	Aldec, “High Performance Computing – Solutions”
1029	JEDEC Dictionary of Terms for Solid-State Technology, 7th Ed.
1030	IEEE 1532-2002 (“Standard for In-System Configuration of Programmable Devices”)
1031	U.S. Patent App. Pub. No. 2016/0049941A1 (“How”)
1032	Cisco, “High Speed Serial Interface (HSSI) Design Specification”
1033	PCI-SIG, “PCI Express Base Specification Revision 1.0”
1034	U.S. Patent App. Pub. No. Ex-1034 2011/0090640A1 (“Samson”)

The '943 patent is directed to a class of device, called programmable logic devices ("PLD"), where the user can alter the logical connections of hardware circuits. The patent recites the concept of locking the PLD to prevent access to user-configurable sections. However, as explained below, PLDs had been widely used for decades, including in server environments that locked PLDs to prevent unauthorized changes.

I. BACKGROUND

PLDs are integrated circuit ("IC") devices that can be configured after manufacturing by altering the connection pattern of logic gates, in contrast to microprocessors with fixed circuits that users cannot reconfigure. PLDs have been widely used for decades. Ex-1010, Abs, 000002-3. For example, a popular type of PLD, called an FPGA, was taught in undergraduate EE courses and textbooks by the 2000s. Ex-1003, ¶¶3-5.

FPGAs contain programmable logic blocks and interconnects. Ex-1010, 000004. By loading configuration data specifying which logic functions and connections were to be activated (and deactivated), users specified the logical interconnections of circuits within the FPGA to create custom hardware to perform custom functions. Ex-1010, 000009-11; Ex-1008, 1:30-37. The interconnects function as a network to relay signals between programmable logic blocks and I/O

circuits that interface with external circuitry or devices, such as a PCIe interface.

Ex-1003, ¶6, Ex-1011:

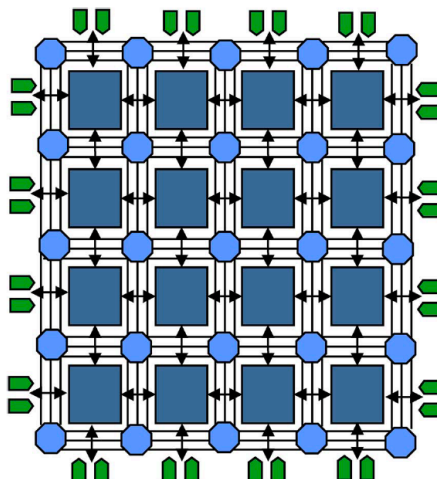


Fig. 4. Generic array FPGA architecture. 4×4 array with three wiring tracks per row and column. Switches are at the circles at intersections. Device inputs and outputs are distributed around the array.

PLDs, including FPGAs, generally included two regions: one (in blue below) with programmable logic blocks and another (in red) for control functions and input/output. *E.g.*, Ex-1004, 10:36-47, Fig. 2; Ex-1006, Fig. 9:

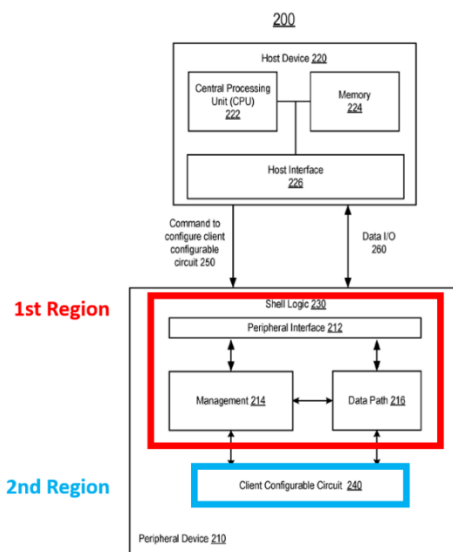


FIG. 2

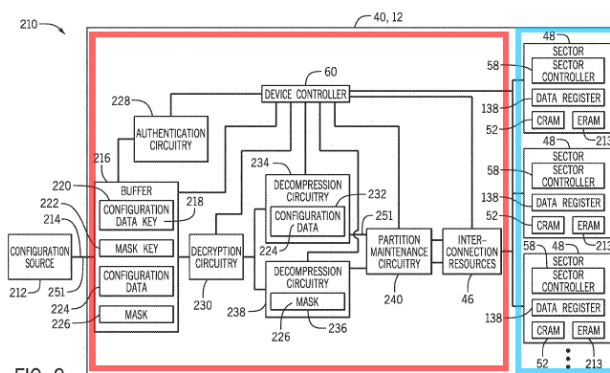


FIG. 9

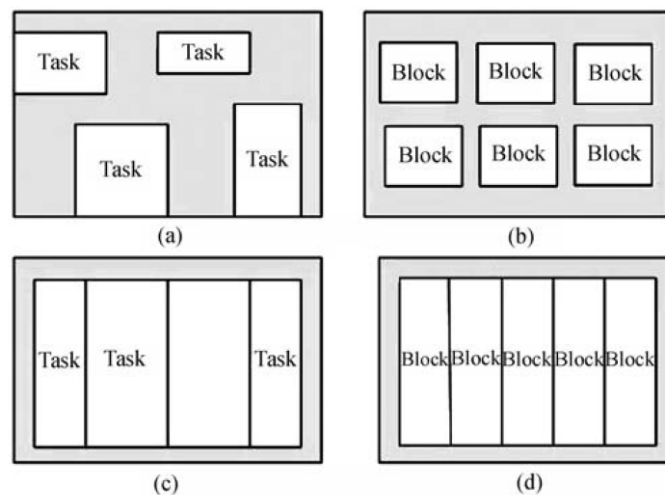
I/O typically included PCIe. Ex-1004, 10:48-65; Ex-1006, ¶30; Ex-1003, ¶11.

Below are examples of FPGA cards plugged into PCIe slots. Ex-1027; Ex-1028;

Ex-1003, ¶¶7-8.



The programmable portion of an FPGA could be reprogrammed by writing configuration data for the entire FPGA or by partial reconfiguration, which “change[d] the configuration of part of [a] FPGA dynamically without disturbing other parts.” Ex-1020, 000001. This was commonly used to divide the programmable fabric into independently-managed partitions. Ex-1020, 000001-2:



Ex-1003, ¶9.

Partitioning was particularly useful for FPGAs in servers because it allowed multi-tenancy, “sharing a single physical device among multiple users” (Ex-1025, 000004), who each loaded their own logic design. Ex-1026, 000003; Ex-1024, 000002-4, Fig. 4; Ex-1023, 000003-5. Dynamic resizing of partitions was widely known. Ex-1020, 000005-8; Ex-1003, ¶10.

The proliferation of cloud computing in the mid-2000s created demand for scalable computing, often using server blades in racks. Ex-1014, 00009-15. PLDs became popular accelerators in data center blades (Ex-1014, 00022; Ex-1017, 000087; Ex-1015, 000002-3), particularly for computation-intensive data flow processing, such as network data, multimedia, big data, and machine learning (Ex-1012, 000021-7). FPGAs were a popular solution for these types of data flow processing. Ex-1012, 000065-67; Ex-1003, ¶¶14-17.

There was a recognized need for security when PLDs were used in networked settings where multiple users had access. Ex-1012, 00100; Ex-1013, 000001-2. Simply locking the configurable portions of the PLD to prevent unauthorized changes was a common approach among a variety of security measure. Ex-1004, 6:33-41 (“isolate a configuration access port so that the customer logic cannot cause the client configurable circuit to be reprogrammed”); Ex-1013, 000002-8; Ex-1012, 00100-101. PLD vendors, including Intel/Altera,

included features to “protect against attempts of partial configuration that inappropriately modify wrong locations.” Ex-1006, ¶¶66-69. This included, for example, using a key for authentication. Ex-1006, ¶¶59-62; Ex-1013, 000003-5; Ex-1012, 00101. Ex-1003, ¶12.

II. THE '943 PATENT

The '943 patent describes a “programmable device for data flow processing in a user-configurable server with swappable pods or cards.” Ex-1001, 1:23-44. The patent does not purport to invent PLDs or data flow processing. It does not provide details regarding how PLDs are designed or manufactured. Rather, it recites PLDs, uses FPGAs for its embodiments (Ex-1001, 3:8-12), and notes the existence of prior art “systems [] appropriate for many data flow and data processing applications.” *Id.*, 1:7-21; Ex-1003, ¶18.

The '943 patent purports to invent a PLD with two regions, where the first region includes functionality that is lockable/unlockable to enable user access to programmable electronic circuits in the second region. Ex-1003, ¶19; Ex-1001, Claim 1, Figs. 3-4:

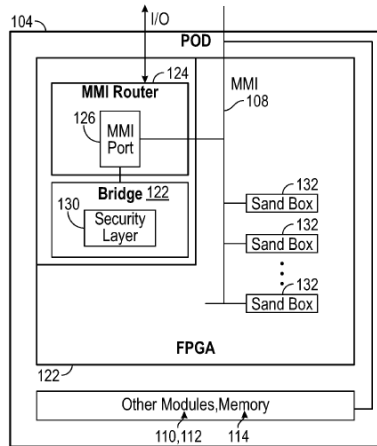


FIG. 3

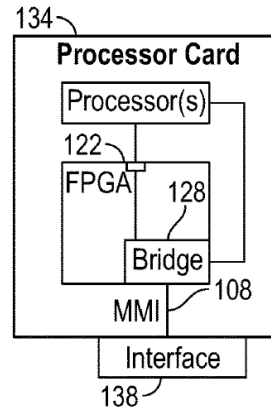


FIG. 4

However, as explained above, PLDs were widely used in data centers for data-flow processing and included security features to prevent unauthorized access by locking and unlocking user-programmable portions. *Supra* §I; Ex-1003, ¶20.

A. Prosecution History

The '943 patent was rejected in one Office Action. In response, the Applicant argued the cited art did not disclose a PLD, which is “a term of art referring to a specific type of integrated circuit that has user programmable electronic circuits.” Ex-1002, 000130-132. The Examiner found this “persuasive.” *Id.*, 000143-144. The patent issued after an Examiner’s amendment to add the last limitation of locking/unlocking access to the sandboxes. Ex-1002, 000141-142. The Applicant argued that this limitation had written description support based on the specification’s discussion of Figure 3. Ex-1002, 000129-130; Ex-1003, ¶¶21-27.

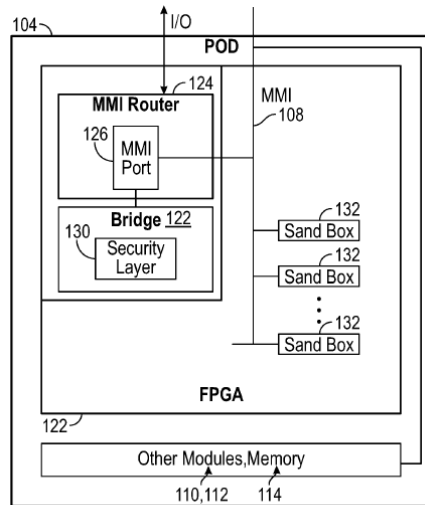


FIG. 3

B. Priority

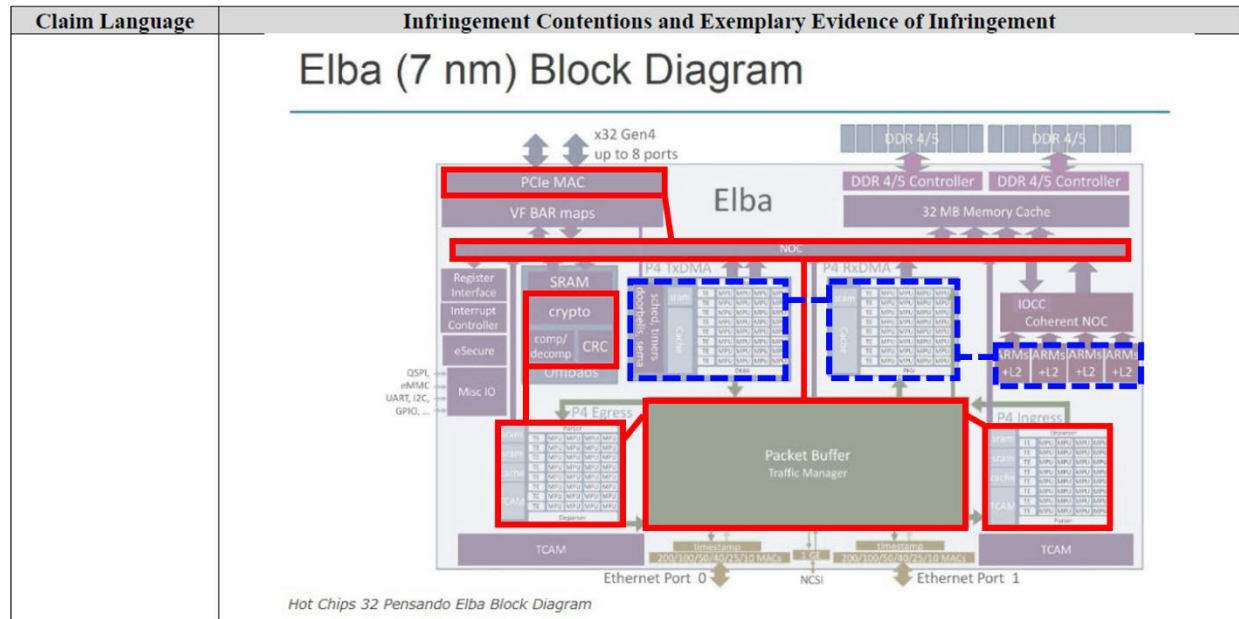
Petitioner demonstrates invalidity as of July 15, 2018. Ex-1003 ¶28.

C. PO's Allegations

In district court, PO accused special-purpose processors that run software written in the P4 programming language, taking the position that any type of processor is a PLD. Ex-1021, 000002-15. PO additionally accused ARM CPU cores of being the claimed sandboxes of a PLD. Ex-1021, 000027; Ex-1003, ¶30.

For the claimed first and second regions, PO created arbitrary groupings of chip components, summarized by the below annotations:¹

¹ Annotations added to figures unless otherwise noted.



See Ex-1021, 000015, 000037-38; Ex-1003, ¶31.

PO also took the view that the claimed router, bridge, and port can be commensurate in scope. For limitation [1b], PO cited various IC components as being in the alleged first region (Ex-1021, 000020) and stated that “[e]ach of these, *separately and together*^[2], constitute the claimed ‘hardware-based or firmware-based router’ and ‘port.’” *Id.* PO then cited a packet buffer as the alleged bridge (Ex-1021, 000033, 000047-49) and accused a “hardware based firewall at the PCIe interface” of also being the bridge because it “prevent[s] unauthorized access” (Ex-1021, 000037-38, 000052-55). Ex-1003, ¶32.

² All emphasis/annotation added unless otherwise noted.

For [1d], PO alleged that the accused product includes a lockable bridge because “the user must log in” to verify account permissions before writing a software image to the card. Ex-1021, 000034-35, 000055-56. PO further cited “various hardware security features to prevent unauthor[iz]ed access.” Ex-1021, 000037-41.

PO alleged that claim 10 was satisfied for the same reasons cited for claim 1. Ex-1021, 000074-108; Ex-1003, ¶¶33-35.

III. FIELD AND LEVEL OF SKILL

The field of the patent is PLDs and data processing systems. That is the focus of claims and specification and is how the patent describes its field and related art. Ex-1001, 1:6-44, 2:47-3:20, 4:27-47, 6:37-45, 5:25-38; Ex-1003, ¶36.

A POSITA would have a bachelor’s degree in electrical engineering or a similar major with at least two years of experience in the field; or a master’s degree with specialization in the field. A person with less education but more relevant practical experience may also meet this standard. Ex-1003, ¶¶37-38.

A POSITA with this background would have had the knowledge and skills to:

- Design distributed systems for data flow processing;
- Deploy hardware/software for streaming, networking, and data processing;
- Understand IC and PLD architectures;

- Design and implement PLDs including FPGAs;
- Develop and configure logic circuits in PLDs/FPGAs;
- Program software/firmware for microprocessors;
- Implement known algorithms as electronic circuits or software; and
- Interface between bus protocols and communication systems.

Ex-1003, ¶39.

IV. CLAIM CONSTRUCTION

A. “programmable logic device”

This construction is relevant to PO’s allegations, but for this IPR, each ground teaches a PLD irrespective of whether the construction is adopted.

“Programmable logic device” means an IC where the user can alter the connection pattern of the circuits. This is confirmed by the claims, specification, and file history (Ex-1004, 4:30-47, 3:8-12; Ex-1002, 000130-132) and the conventional meaning of PLD (Ex-1029, 000018; Ex-1029, 000018; Ex-1030, 000014). The ’943 patent identifies FPGAs as a type of PLD (Ex-1004, 4:30-35, 3:8-12, 1:23-49) while reciting PLDs in general. Ex-1003, ¶¶42-49.

V. INVALIDITY

Petitioner requests *inter partes* review and cancellation of the challenged claims on the following grounds:

Grounds	Claims	Statutory Basis	Prior Art ³
1	1-5, 10-14	§103	Seshadri
2	1-5, 9-14, 18	§103	Seshadri and Biederman
3	1-5, 10-14	§103	Atsatt-113
4	1-5, 9-14, 18	§103	Atsatt-113 and Biederman
5	6, 15	§103	Grounds 1-4 and Tarafdar
6	7, 16	§103	Grounds 1-4 and Kulkarni
7	8, 17	§103	Grounds 1-4 and Atsatt-923

The Grounds render the challenged claims obvious because any differences between claimed subject matter and prior art are such that the subject matter, as a whole, was obvious to a POSITA. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007).

A. Grounds 1-2

The challenged claims are obvious based on Seshadri. The combination with Biederman teaches using/resizing FPGA partitions for claims 9-10 and 18. *Infra* §V.A.4, §V.A.12 (subsections explaining motivation to combine). Ex-1003, ¶¶51-148.

³ AIA §102 applies.

1. U.S. Patent 10,776,142 (“Seshadri”)

Seshadri is §102(a)(2) prior art filed December 13, 2017. Seshadri is directed to an FPGA architecture for use in distribute, scalable computing environments that partitions the programmable fabric into a shell logic region and a “client configurable circuit” region. Ex-1004, 10:36-49, 10:7-11, 10:36-47, 2:60-3:27:

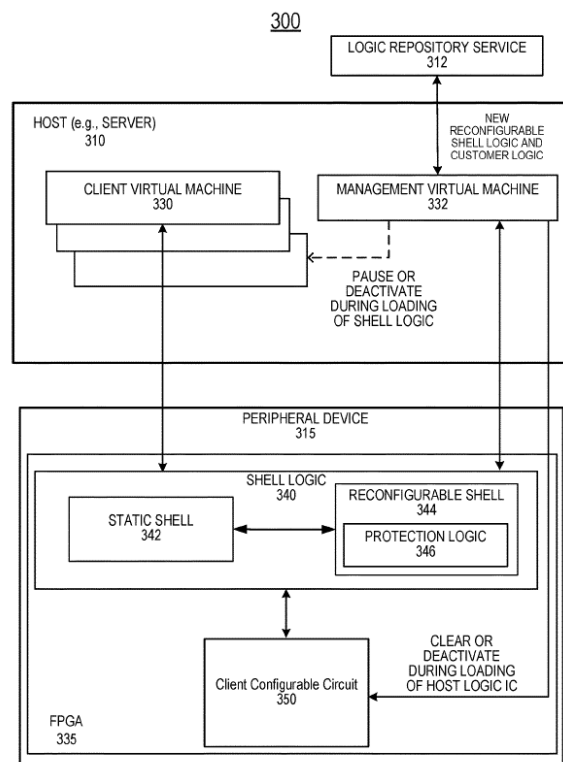


FIG. 3

The shell logic “wrap[s] or encapsulat[es] the client configurable circuit[,]” provides “bridging functions,” and checks user account permissions before reconfiguring client-configurable circuits. Ex-1004, 3:41-64, 9:48-10:15, Fig. 6; Ex-1003, ¶¶54-56.

Seshadri is analogous art in the same field of endeavor and reasonably pertinent to the '943 patent (*supra* §III), because it proposes an architecture for processing streams of data with FPGAs in a server environment. Ex-1004, 4:2-11, 26:27-61, 20:67-21:4, Fig. 1; Ex-1003, ¶57.

2. U.S. Patent Application Publication No. 2018/0191642A1 ("Biederman")

Biederman is §§102(a)(1)-(2) prior art filed December 30, 2016, published July 5, 2018. Biederman is directed to processing data streams, distributing packets among the partitions of and FPGA and dynamically resizing partitions and choosing "program to load into" the partitions. Ex-1005, ¶134, Abs., ¶¶27-30, Figs. 10, 8:

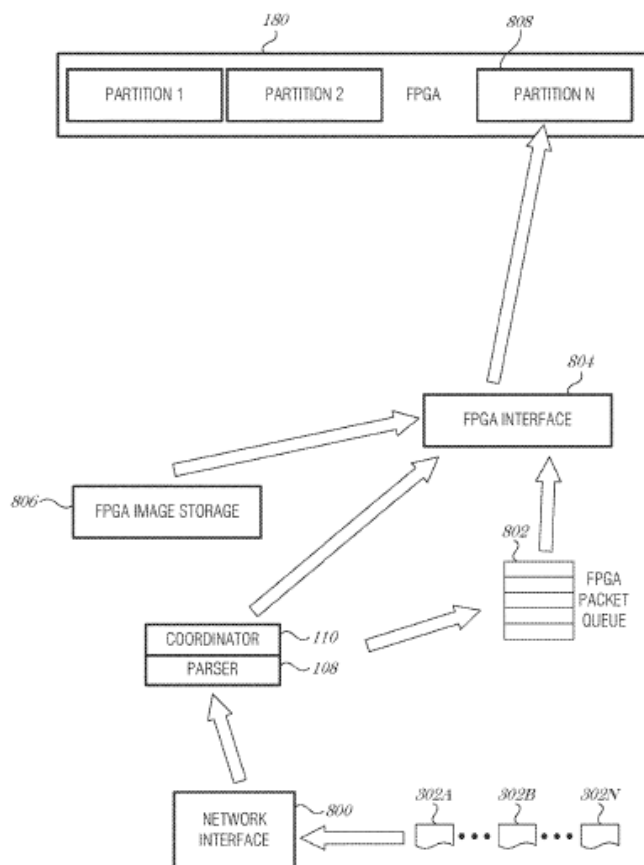


FIG. 8

Biederman applies its teachings for server-based data flow processing services such as software defined networking, network function virtualization, virtual switches, and data centers. Ex-1005, ¶¶174-177, ¶¶182-184. Ex-1003, ¶60; Ex-1003, ¶¶59-61.

Biederman is analogous art in the same field of endeavor and reasonably pertinent to the '943 patent (*supra* §III), because it discloses an architecture for processing network data, including video streams and contemplates using multiple

reconfigurable FPGAs in a server for data flow processing. Ex-1005, Abs., ¶134; Ex-1003, ¶62.

3. Claim 1

[1pre]. A programmable device for data flow processing in a user-configurable server with swappable pods or cards, comprising:

Seshadri teaches the preamble. Ex-1003, ¶¶63-68.

For example, Seshadri teaches “a programmable circuit (e.g., an FPGA)” (Ex-1004, 2:44-3:11, 10:36-65) used in a “configurable computing system” with “configurable devices” allowing remote users to load custom logic on the FPGAs to configure the server (Ex-1004, Abstract, 13:40-14:6, 20:67-21:4). This includes servers providing “networked computing resources.” Ex-1004, 1:7-22, 13:40-48, Figs. 1-2, 4, 8:

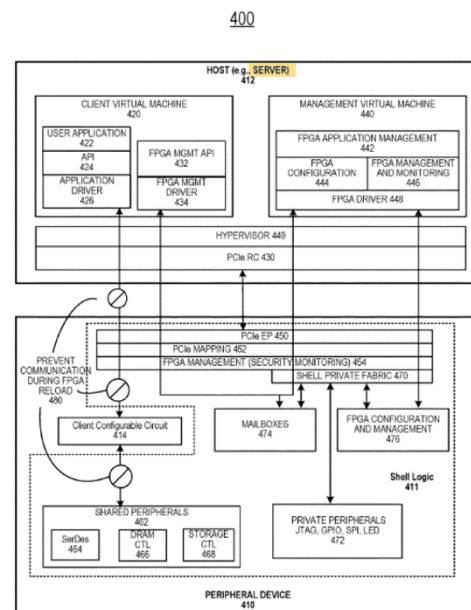
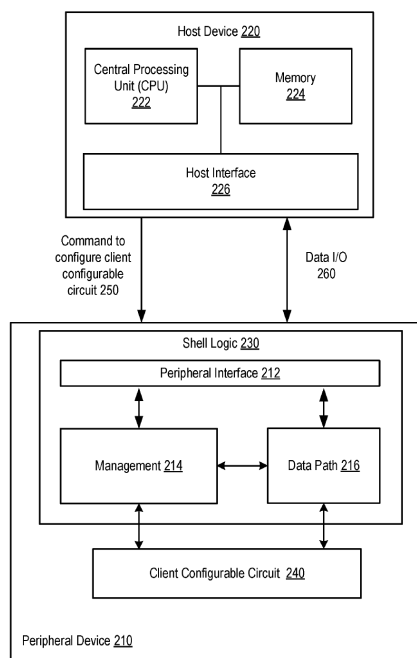


FIG. 4

Ex-1003, ¶¶64-65.

Seshadri's servers are equipped with swappable "plug-in cards" that have "programmable ICs" including FPGAs. Ex-1004, 13:40-48, 2:60-3:11. Therefore, users configured servers by swapping out FPGA cards and loading custom logic. *Id.*, Abs., 13:40-14:6; Ex-1003, ¶66.

Seshadri uses servers and FPGAs for data flow processing, loading user logic "to perform data streaming"⁴ using "different customer logics." Ex-1004, 4:2-9. Seshadri teaches cloud computing and Internet applications, which process flows of data in packets. *Id.*, 1:7-31. Ex-1003, ¶67.

[1a] a programmable logic device (PLD) having a first region and a second region;
--

Seshadri teaches [1a]. Ex-1003, ¶¶69-73.

Seshadri discloses "programmable integrated circuits (ICs)" that have a first "shell logic" region and a second "client configurable circuit" region. Ex-1004, 10:36-47, 2:44-3:11, Figs. 2, 3-5; Ex-1003, ¶69.

⁴ The '943 patent describes "processing stream live data" as an example of data flow processing. Ex-1001, 11:42-12:42, 3:50-64, Figs. 1B, 10.

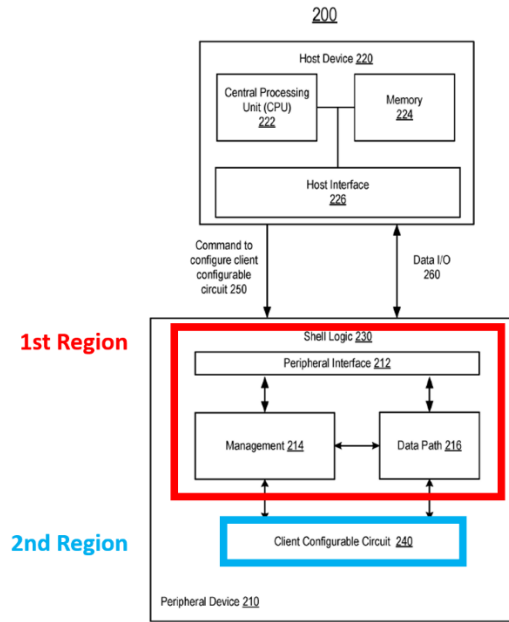


FIG. 2

Seshadri’s figures illustrate various teachings consistent with the configuration of Figure 2. Ex-1004, 1:40-51, Figs. 2-5. “FIG. 2 is a simplified diagram” of the host device and FPGA. *Id.*, 6:42-52. Figures 3-4 show further details, while Figure 5 illustrates functional blocks within the shell logic for continuous operation during reconfiguration. *Id.*, 10:16-11:30, 13:40-47, 16:61-17:15. These teachings are complementary, overlapping, and meant to be used together. As Seshadri explains, the figures “include the software and hardware components described above with respect to” the other figures, even if not shown. *Id.*, 17:16-36; Ex-1003, ¶70.

Seshadri teaches PLDs: “programmable circuit (e.g., an FPGA).”⁵ Ex-1004, 2:44-3:11, 10:36-47; Ex-1003, ¶71.

Seshadri also teaches the PLD is an IC where the user can alter the connection pattern of the circuits. *Supra* §IV.A (claim construction). Seshadri teaches an FPGA (Ex-1004, 10:36-47, 13:40-48, 20:46-48, Fig. 3) and explains that the FPGA’s programmable circuits can be altered to perform a logic function specified by the user, meaning the connection pattern of the circuits is modified to implement the user’s logic function. Ex-1004, 2:44-3:11. Ex-1003, ¶72.

[1b] the first region comprising a hardware-based or firmware-based router with a port enabling communication between the router and the second region; and

Seshadri teaches [1b]. Ex-1003, ¶¶74-79. Seshadri teaches the first region comprising a router with a port.⁶ Ex-1004, Fig. 2:

⁵ The ’943 specification uses FPGAs as the PLD. Ex-1001, 4:27-5:49, 13:14-16, Figs. 3-4.

⁶ PO alleged various IC components “separately and together” constitute the router and port. Ex-1021, 000020; *supra* §II.C.

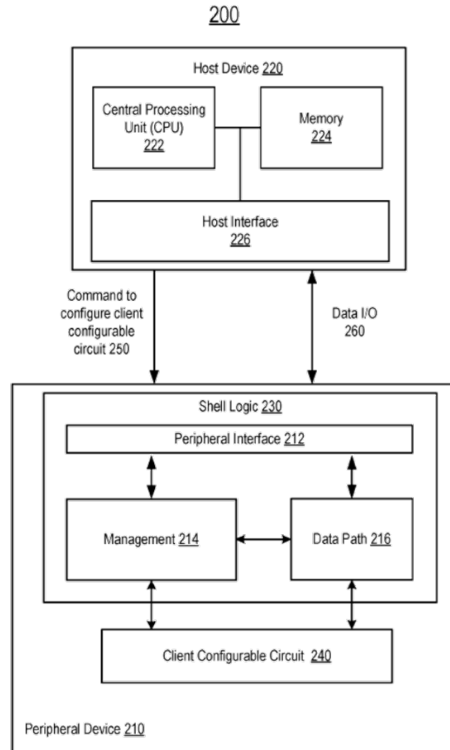


FIG. 2

Ex-1003, ¶74.

Regarding the router, the shell logic region includes functionality that “perform[s] bridging functions between the local interconnect (e.g., the PCIe interconnect) and the client configurable logic.” Ex-1004, 6:11-41. This functionality (including but not limited to management functions and the data-path) routes communication between the shell logic and the client-configurable circuit in order to convey messages to/from the host device. Ex-1004, 8:37-61 (“Data path module 216 in shell logic 230 may be used to exchange information (e.g., data input/output 260) between host device 220 and peripheral device 210...”), Fig. 2. Inbound communication from the host device is received over the

peripheral interface, and functionality in the shell logic “route[s] commands to functional modules or blocks of peripheral device 210 based on an address.” Ex-1004, 7:56-8:8, 14:37-53, Fig. 4. Outbound communication is passed from client-configurable circuits to the peripheral interface that “us[es] a command including an address associated with host device 220.” *Id.* The data-path module thus routes communications between the first (shell logic) region and the second (client-configurable circuit) region so the client virtual machine (VM) can use the accelerator logic in the client-configurable circuit. *Id.*, 5:32-57, 6:43-51, 12:32-49, 16:3-10; Ex-1003, ¶75.

To facilitate routing of messages with the second (client-configurable circuit) region, Seshadri’s routing functionality further includes a port that enables communication with the second region. This includes, for example, management functions and the data-path, such as mailbox functionality through which messages are passed between the control plane (including shell logic) and data plane (including client-configurable circuits). Ex-1004, 14:63-15:8 (“*[M]ailboxes 474 may be used to pass messages and other information between the control plane and the data plane.*”), 12:32-49 (“*The control plane functions may include shell logic 340.... a data plane function may be used to accelerate a client's application that is running on host 310.*”). The mailboxes comprise part of the data-path because they provide communication functionality.

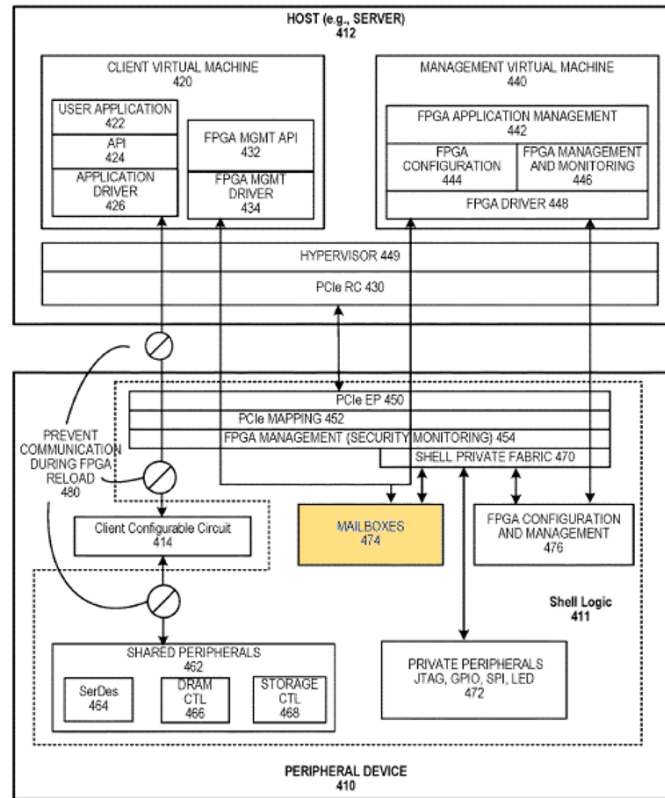


FIG. 4

Ex-1003, ¶76.

In summary, Seshadri teaches a port that comprises the functionality in the shell logic for enabling communication with client configurable circuits. The port also comprises a bridge, explained for [1d]. Seshadri teaches a router that comprises the shell logic port and other functionality in the shell logic for bridging and routing communications. Ex-1003, ¶77.

The router in the shell logic is hardware-based because it is implemented in the hardware programmable fabric⁷ of the FPGA. Ex-1004, 2:2-5, 3:24-27; Ex-1003, ¶78.

[1c] the second region comprising one or more sandboxes with user-definable programmable electronic circuits of the PLD,

Seshadri teaches [1c]. Ex-1003, ¶¶80-85. Seshadri teaches “sandboxing” of a user’s client-configurable circuits (in the second region) to “isolate” it from other clients’ custom logic. Ex-1004, 10:66-11:16 (“Shell logic 340 may provide encapsulation or sandboxing of client configurable circuit 350 so that one client may not obtain secure information associated with the operation of another client's customer logic.”), 2:44-59, 6:11-41. Therefore, Seshadri teaches user sandboxes that are isolated from one other. *Id.*; Ex-1003, ¶81.

The client-configurable circuit comprises one or more sandboxes with programmable circuits that the user defines by loading a custom specification to “implement a customer logic for the client.” Ex-1004, 7:22-36, 12:10-31, 3:2-14. These circuits are defined by the user because the user programs the actual circuit

⁷ In columns 2-3, the mention of a “client configurable circuit” refers to the programmable fabric of an FPGA. Ex-1004, 2:2-54, 3:2-14, 3:41-64; Ex-1003, n. 7.

interconnections of PLD/FPGA circuits to create logical functions defined by the user. Ex-1004, 7:22-36, 10:36-65, 12:10-31, 17:16-36 (“[C]lient configurable circuit...may be programmed or otherwise configured to implement *customer defined logic*....”). The client-configurable circuits are electronic circuits in the PLD/FPGA, which is an IC. Ex-1003, ¶¶82-84.

[1d] wherein the port comprises: a bridge being lockable to prevent user access, and unlockable to enable user access, to each of the sandboxes and the user-definable programmable electronic circuits therein.

Seshadri teaches [1d]. Ex-1003, ¶¶86-91. Seshadri’s shell logic includes port functionality that enables communication with the client-configurable circuits. *Supra* §V.A.3[1b]. The port comprises functionality to perform “bridging functions” between the PCIe interface and the client-configurable logic. Ex-1004, 3:41-64, 4:11-41 (“The static shell [logic] may also include circuits...for bridging the communications....”), 6:11-41. For example, the shell logic’s “mailbox 588 may be used as a bridge[.]” Ex-1004, 17:37-57, 18:4-24 (“mailbox 588 for bridging the communications”):

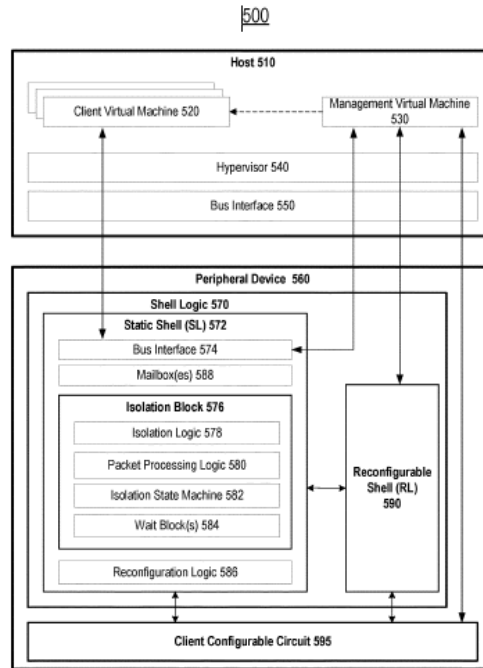


FIG. 5

Ex-1003, ¶87.

The bridge is lockable/unlockable to prevent or enable user access to each sandbox. For example, the shell logic includes functionality to “isolate a configuration access port”; this functionality is locked/unlocked to disallow or allow configuration of client-configurable circuits. Ex-1004, 6:11-41, 12:10-31; Ex-1003, ¶88.

When configuration is requested by the user, the shell logic checks “customer account permissions”⁸ to reject or accept the request to unlock the reprogramming functionality. Ex-1004, 9:48-10:15 (“...Subsequently...the customer logic can be configured on the client configurable circuits and activated using the reconfigurable shell.”). Seshadri locks/unlocks the bridge to control user access to programming and “increase the security, performance, privacy, and/or availability of the computing resources” in the FPGA. Ex-1004, 3:41-64; Ex-1003, ¶89.

In summary, Seshadri locks the bridge to prevent users from rewriting the FPGA and unlocks to enable user access to reconfigure sandboxed portions, including when the entire client-configurable circuit is a single sandbox. Ex-1004, 9:48-10:15, 6:11-41, 12:10-31. This bridging functionality is part of the bridge on the port because it is the portion of the shell logic that enables/disables communications for reconfiguring the client-configurable circuits. *See id.* Moreover, this would have been obvious because Seshadri teaches that the shell logic includes routing functionality with a port for enabling communications to the

⁸ PO alleged [1d] is satisfied when the “the user must log in” to verify account permissions before writing an image. Ex-1021, 000034-35, 000055-56; *supra* §II.C.

client-configurable circuits (*supra* §V.A.3[1b]), and that is what the bridging functionality performs for reconfiguration communications. *See id.* The client-configurable circuit comprises user-definable programmable electronic circuits. *Supra* §V.A.3[1c]; Ex-1003, ¶90.

4. Claim 10

[10pre]-[10a] ...

[10pre]-[10a] are recited in [1pre]-[1a]. They are satisfied for the same reasons. *Supra* §V.A.3[1pre]-[1a] (incorporated by reference); Ex-1003, ¶93.

[10b] the first region comprising...
--

[10b] overlaps with [1b]. For both claims, the first region comprises a router with a bridge and port.⁹ Ex-1003, ¶94.

⁹ The '943 patent uses router and bridge interchangeably. Figure 3 shows the *router* providing communication between internal/external modules and having external and MMI ports. Ex-1001, 4:52-59, 3:8-12. Elsewhere, the specification describes the *bridge* as having these features. Ex-1001, 4:52-59. Claim 1 recites a router with a “port compris[ing] a bridge”; claim 10 recites a router having a “bridge with a port”; but the specification discloses the bridge comprising a router (Ex-1001, 3:8-12) while depicting the bridge external to both router and port (Fig.

As explained above, Seshadri teaches the first region comprising a hardware-based router with a bridge and port that enables communication between the router and the client configurable circuits. *Supra* §§V.A.3[1b], [1d] (incorporated by reference); Ex-1003, ¶95.

[10c] the second region comprising a plurality of sandboxes...

[10c] recites overlapping language with [1c] that is satisfied for the same reasons. *Supra* §V.A.3[1c] (incorporated by reference). [10c] additionally recites a plurality of sandboxes. Ex-1003, ¶¶96-104.

Seshadri teaches a plurality of sandboxes.

Seshadri teaches sandboxing multiple users' custom logics in a "multi-tenant cloud environment" (Ex-1004, 2:63-4:2, 10:66-11:16) to prevent one user's logic from interfering with another's (*id.*, 3:41-64). Seshadri refers to "client configurable circuits" in the plural. *Id.*, 9:48-53, 10:8-14. Figure 1 shows multiple VMs corresponding to multiple client configurable circuits. Ex-1003, ¶97.

3). In district court, PO alleged that various IC components "separately and together" constitute the router and port (Ex-1021, 000020) and relied on the same support for [1b] and [10b] (Ex-1021, 000079-84, 000020-26). *Supra* §II.C

Biederman further teaches **a plurality of sandboxes.**

Biederman teaches “multiple partitions” in the FPGA’s programmable fabric hosting a plurality of logic designs. Ex-1005, ¶129:

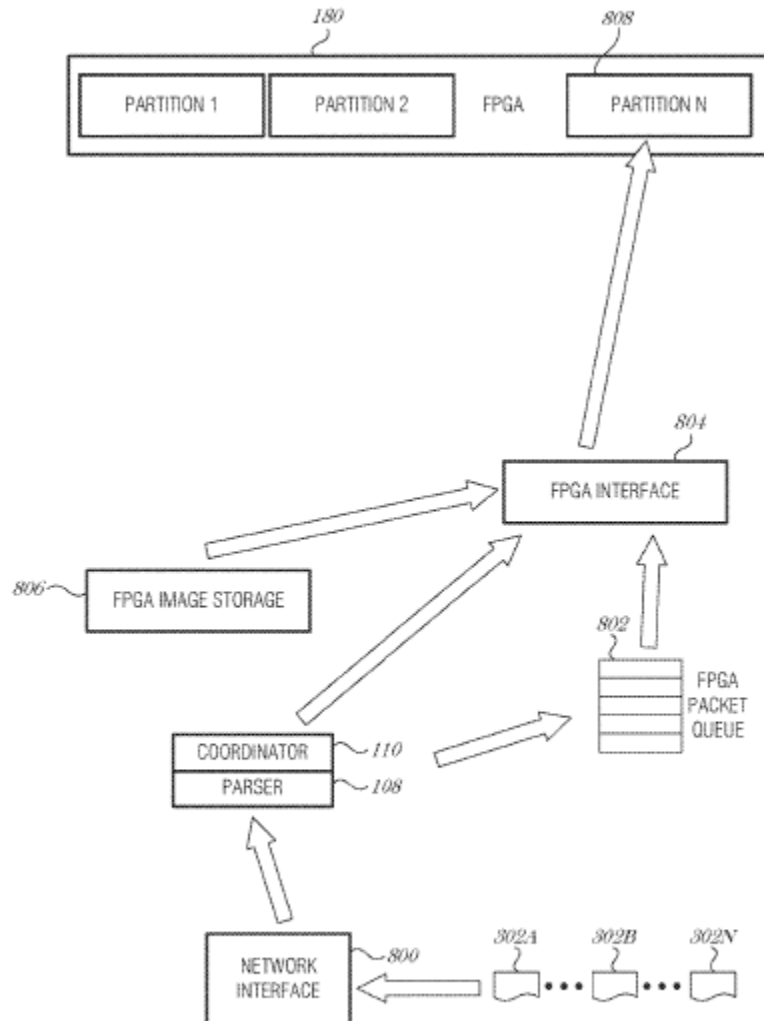


FIG. 8

Like Seshadri’s shell logic, Biederman’s coordinator circuit manages the partitions, determining which logic design is loaded in each partition. Ex-1005, ¶¶123-128; Ex-1003, ¶98.

Ground 2 divides Sheshadri's client-configurable circuit into multiple partitions by applying Biederman's teaching of multiple partitions to the circuits. *See* Ex-1005, ¶¶123-129, Fig. 8. It was obvious to partition the client-configurable circuits to allow multiple tenants to each use different areas in a single FPGA because Seshadri already teaches partitioning the configurable areas of a PLD into different areas for different logic circuits. Ex-1004, 10:36-49, 10:7-11; Ex-1003, ¶99.

Motivation to Combine. A POSITA would have been motivated to combine Seshadri's and Biederman's teachings because it would have combined prior art elements according to known methods to yield predictable results. Biederman teaches a known technique for partitioning the programmable fabric of an FPGA. Ex-1005, ¶129. Seshadri teaches a known FPGA configuration where the programmable fabric is partitioned into a shell area and a client-configurable circuit area, which can be further partitioned. It was well-known to partition an FPGA's programmable fabric into different portions for different users. *Supra* §I; Ex-1026, 000003; Ex-1007, 000003; Ex-1024, 000002-4; Ex-1023, 000003-5. Therefore, there is express motivation for the combination, which yields the predictable result of partitioning the client-configurable circuit into distinct areas for customer logic from different users. Ex-1003, ¶100.

The combination has several benefits. It serves more users with a smaller number of FPGAs, providing cost savings for cloud providers and reducing the entry barrier. It improves performance by reducing assignment switching of FPGAs. Ex-1003, ¶101.

A POSITA would have had a reasonable expectation of success when combining Seshadri and Biederman. The combination would not have changed the principle of operation for either reference. Seshadri would still be used to sandbox client-configurable circuits in an FPGA; Biederman would still be used for partitioning. The operation of the programmable logic sectors, interconnects, and other FPGA elements would operate the same way. Ex-1003, ¶102.

Seshadri and Biederman provide complementary teachings for using FPGAs in servers. Biederman's FPGA-partitioning details would naturally have been compatible with Seshadri's FPGA teachings, which already partition the FPGA into several sections, as was known in the art. Ex-1005, ¶129, Fig. 8; Ex-1004, 10:7-49, 9:48-53; *supra* §I. A POSITA with 2 years of experience would have been fully capable of making the combination. *Supra* §IV; Ex-1003, ¶103.

[10d] an internal communication network in the PLD, coupling the port of the bridge of the router and the one or more sandboxes for the data flow processing,

Seshadri teaches [10d]. Ex-1003, ¶¶105-108. For example, Figure 2 shows the data-path serving as a network for internal communication between various modules within the FPGA:

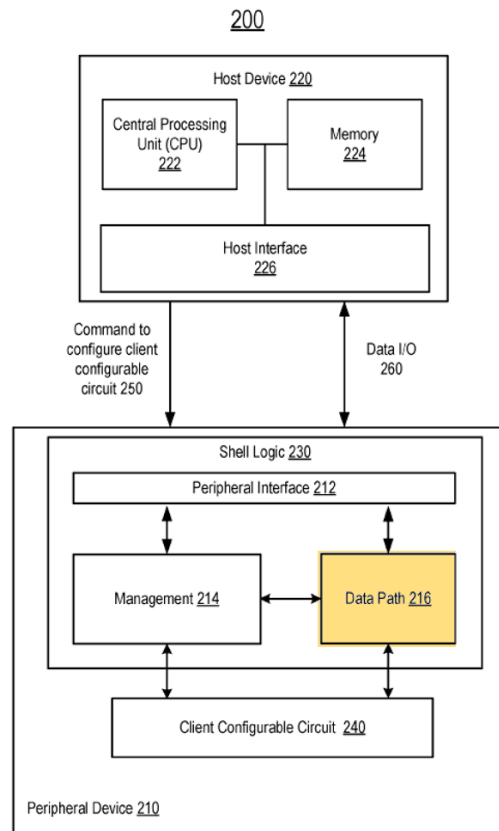


FIG. 2

The port of the bridge of the router includes the portions of the shell logic that enable communication with the client-configurable circuit region. *Supra* §V.A.3[1b], §V.A.4[10b]. The data-path couples the port to the client-configurable circuits. Ex-1004, Fig. 2; Ex-1003, ¶106.

Figure 2 depicts the data-path as a connection fabric—a path for data—which Seshadri teaches is a network (Ex-1004, 26:1-3 (“‘fabric’ and ‘network’ may be used interchangeably”). The data-path is a network that connects internal FPGA components for communication in the FPGA. FPGAs comprise interconnect resources that serve as a fabric for relaying internal communication from one part of the IC to another; this was a basic concept of FPGAs. *Supra* §I; Ex-1010, 000004, 000176-177; Ex-1012, 000022; Ex-1006, ¶¶30-34; Ex-1008, 1:22-29; Ex-1011, 000003-4; Ex-1009, ¶41-42. A POSITA would have understood Seshadri’s teachings regarding a data-path to encompass the interconnect resources that form an internal communication network of the FPGA. Ex-1003, ¶107.

Seshadri’s sandboxes are for data flow processing. Seshadri loads customer logic into the sandboxed client-configurable circuits to perform customer applications, which include data flow processing. *Supra* §V.A.3[1pre], §V.A.3. [1c]; Ex-1003, ¶105.

[10e] wherein the bridge is lockable to prevent user access, and unlockable to enable user access, to the plurality of sandboxes and user-definable programmable electronic circuits therein.

[10e] recites overlapping language with [1d] that is satisfied for the same reasons.¹⁰ *Supra* §V.A.3[1d] (incorporated by reference); Ex-1003, ¶¶109-111.

[10e] additionally recites a plurality of sandboxes, which is taught by Grounds 1-2. *Supra* §V.A.4[10c] (incorporated by reference). Seshadri's teachings of locking/unlocking the bridge apply equally to multiple sandboxed partitions. Seshadri checks user account permissions to lock/unlock the sandboxed client-configurable circuits because it contemplates multiple sandboxed client-configurable circuit areas in a "multi-tenant cloud environment." Ex-1004, 2:63-4:2, Figs. 8, 1. Therefore, Seshadri's teachings for [1d] apply to a plurality of sandboxes on the same FPGA. Ex-1003, ¶110.

5. Claim 2

2. The programmable device of claim 1, further comprising: the router communicative to translate between a first protocol and a second protocol, for messaging with system resources.

Seshadri teaches claim 2. Ex-1003, ¶¶112-117. Seshadri's router functionality in the shell logic comprises a mailbox and data-path "interposed

¹⁰ PO relied on the same support for [1d] and [10e] in district court. *Supra* §II.C; Ex-1021, 000099-108, Ex-1021, 000032-41.

between client configurable circuit 240 and the physical interconnect (e.g., the PCIe bus),” translating between the FPGA’s internal format and the PCIe bus protocol. Ex-1004, 8:37-61, Abs. The data-path module functions “as a translation layer,” . “receiving information from client configurable circuit 240, and...format[ing] the information for transmission from peripheral interface 212...to meet protocol specifications.” Ex-1004, 8:37-61; Ex-1003, ¶113.

Messages sent from client-configurable circuits to the rest of the system are translated between a first protocol of the peripheral interface (e.g., PCIe) and the FPGA’s internal protocol. Ex-1004, 6:11-41 (“The shell logic may also be responsible for forming packets or bus transactions on the local interconnect [(PCIe interface)] and ensuring that certain protocol requirements are met.”), 9:12-22 (“a mailbox function”). PCIe was a well-known serial-protocol designed for inter-device or inter-chip communication; it was not designed or used for internal chip communications. Ex-1033, 000037 (PCIe standard explaining “parallel-to-serial and serial-to-parallel conversion” for translating to internal “implementation-specific format”); Ex-1003, ¶114.

Seshadri teaches an FPGA with various internal components that communicate through internal buses. *E.g.*, Ex-1004, 14:54-63, Fig. 4 (“shell private fabric 470”). FPGAs included internal buses and fabrics to allow different portions of the FPGA to communicate with each other; this was a basic aspect of

IC and FPGA design. *Supra* §I; Ex-1010, 000004, 000176-177; Ex-1012, 000022; Ex-1006, ¶30, ¶33-34; Ex-1008, 1:22-29; Ex-1011, 000003-4; Ex-1009, ¶41-42.

Here, Seshadri explains that the data-path module translates between the protocol used by the FPGA's physical interface (e.g., PCIe) and the internal protocol used by the client-configurable circuits. Ex-1004, 8:37-61. Therefore, Seshadri teaches the router functionality of the shell logic being communicative to translate between a first protocol and a second protocol. Ex-1003, ¶115.

The router functionality does this for messaging with system resources external to the FPGA, such as “host systems, processor nodes, storage subsystems” in the distributed computing platform. Ex-1004, 26:4-8, 23:24-42 (e.g., “bus interface module,” “memory,” “network interface module”), Figs. 7-8.

Additionally, Seshadri's FPGAs are meant to allow “[c]lients [to] access the networked computing resources via a network, such as the Internet.” Ex-1004, 1:5-32, 26:36-61; Ex-1003, ¶116.

6. Claim 11

Claim 11 recites overlapping limitations with claim 2 that are satisfied for the same reasons. *Supra* §V.A.5 (incorporated by reference); Ex-1003, ¶¶118-120.

Seshadri further teaches **a first protocol used external to the second region and a second protocol used in the internal communication network in the PLD**. The protocols were explained for claim 2. The first protocol is used

over the external PCIe interface, which is external to the second region (client-configurable circuits). Ex-1004, Fig. 4 (arrows between “PCIe RC 430” and “PCIe EP 450”). The second protocol is the FPGA’s internal protocol for its internal network over interconnects and internal buses; the data-path module translates between the first protocol used at the peripheral interface to a format suitable for the client-configurable circuits, “sized to meet protocol specifications.” Ex-1004, 8:37-61; Ex-1003, ¶119.

7. Claim 3

3. The programmable device of claim 1, wherein each of the sandboxes comprises user-programmable one or more of: analog circuitry, digital circuitry, memory, processors or controllers.
--

Seshadri teaches claim 3. Ex-1003, ¶¶121-124. Seshadri explains that the client-configurable circuit area comprises user-programmable digital circuitry: “hardware that can be programmed or configured to perform a logic function specified by configuration data that is applied to or loaded on the configurable circuit” by a user. Ex-1004, 2:60-3:11; *supra* §V.A.3[1c] (incorporated by reference). These user-programmable circuits are located on FPGAs; therefore, each sandbox comprises digital “logic” programmed by the user. *Id.*, 6:48-7:36, 2:60-4:9, 10:36-47; *supra* §I. Examples include, e.g., “logic elements... clocking resources ... digital signal processing (DSP)...” Ex-1004, 17:16-36. To the extent

PO argues Seshadri's circuitry is not digital, it would satisfy this claim as analog circuitry. Ex-1003, ¶¶122-123.

8. Claim 12

Claim 12 recites overlapping limitations with claim 3 that are satisfied for the same reasons. *Supra* §V.A.7 (incorporated by reference). Grounds 1-2 additionally teach a plurality of sandboxes that comprise digital or analog circuitry as explained for claim 3. *Supra* §V.A.4[10c] (incorporated by reference); Ex-1003, ¶125.

9. Claims 4/13

4. The programmable device of claim 1, wherein the port comprises: a bridge being lockable to prevent reconfiguration of the bridge and prevent reconfiguration of the router while enabling reconfiguration of the one or more sandboxes.

Seshadri teaches claim 4. Ex-1003, ¶¶126-134. Seshadri teaches the port comprising a bridge that prevents reconfiguration of the bridge and router functionality in the static shell logic while enabling reconfiguration of the sandboxes (e.g., client-configurable circuit). Functionality in the shell logic controls reconfiguration of the client-configurable circuit area. Ex-1004, 12:10-31, 11:16-22, 18:35-49, 20:21-22:67 (outlining programming details); *supra* §V.A.3[1d] (incorporated by reference). Outside of FPGA reconfiguration, the shell logic locks ("isolate[s]") the configuration access port to prevent reconfiguration of the client-configurable circuit. Ex-1004, 6:11-41. Thus, the

shell logic unlocks the configuration access port to allow reconfiguration of the client-configurable circuits. *See id.* This would have been further obvious because the FPGA cannot be reconfigured when the access port is locked. *See id.*; Ex-1003, ¶¶127-128.

During reconfiguration, the router, bridge, and port functionality (e.g., mailbox and data-path modules) is in the static portion of the shell (“static shell”). Ex-1004, 4:48-5:3, 18:4-33, 18:63-19:22. The static shell controls access to the configuration access port during reconfiguration because the other portion—the reconfigurable shell—is isolated so it can also be reconfigured. Ex-1004, 18:4-62, 19:23-63. Therefore, the shell logic functionality—including bridging functionality—is in the static portion of the shell during reconfiguration, so that the shell logic can continue to operate and control the FPGA configuration process while reconfigurable portions are being changed. Ex-1004, 18:4-19:63. The reconfigurable shell activates reconfiguration logic in the static shell to take charge of reconfiguration. Ex-1004, 17:58-18:3, 22:51-22:55, Fig. 5:

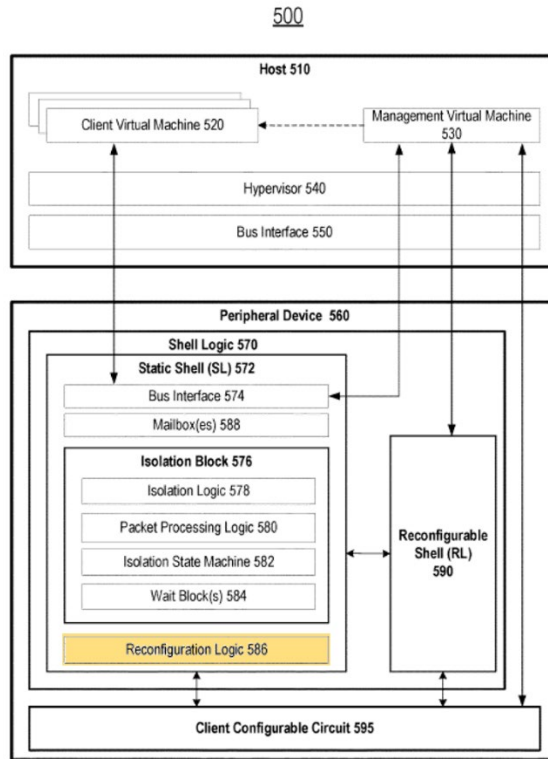


FIG. 5

Ex-1004; Ex-1003, ¶¶129-131.

Seshadri prevents reconfiguration of the bridge and router, which are in the static shell, while the client-configurable circuit area is reconfigured. The static shell can be unlocked for reconfiguration only if “an externally controlled reset signal or a power cycle occurs,” but is otherwise locked from modification. Ex-1004, 11:13-40, 17:37-57. Therefore, the static shell, including the router, bridge, and port functions discussed above, is prevented from being reconfigured when the client-configurable circuit and reconfigurable shell are being reconfigured. Ex-1004, 4:48-5:3. Ex-1003, ¶132.

13.

Claim 13 recites overlapping limitations with claim 4 that are satisfied for the same reasons. *Supra* §V.A.9[4]. Claim 13 additionally requires a plurality of sandboxes, which is taught by Grounds 1-2. *Supra* §V.A.4[10c] (incorporated by reference). Seshadri's teachings for claim 4 equally apply to a plurality of partitions because the static shell controls reconfiguration of the entire client-configurable circuit region. Ex-1003, ¶133.

10. Claim 5

5. The programmable device of claim 1, wherein the port comprises: a bridge having a security layer to authenticate a user or verify a permission, to reconfigure one or more of the one or more sandboxes.

Seshadri teaches claim 5. Ex-1003, ¶¶135-137. For example, the shell logic carries out "security functionality." Ex-1004, 10:66-11:16. Functionality in shell logic verifies that the user's customer account has permission to reconfigure the client-configurable circuits. Ex-1004, 9:48-10:15. After verification, the shell logic then proceeds to reconfigure the requested one or more sandboxes (e.g., client-configurable circuits) by unlocking access to the configuration access port. Ex-1004, 6:11-41, 12:10-31. Grounds 1-2 teach locking/unlocking the bridge to reconfigure the client-configurable circuits. *Supra* §V.A.3[1d] (incorporated by reference), §V.A.9 (incorporated by reference). Ex-1003, ¶¶136-137.

11. Claim 14

Claim 14 recites overlapping limitations with claim 5, which are satisfied for the same reasons. *Supra* §V.A.10 (incorporated by reference). Claim 14 also recites a plurality of sandboxes, taught by Grounds 1-2. *Supra* §V.A.4[10c] (incorporated by reference), to which the teachings for claim 5 apply because both claims require the security layer to act to reconfigure “one or more” of the sandboxes. Ex-1003, ¶¶138-139.

12. Claims 9/18

9. The programmable device of claim 1, wherein the one or more sandboxes comprises a plurality of sandboxes that are logically or physically combinable to form a larger sandbox.

Ground 2 teaches claim 9. Ex-1003, ¶¶140-148. For example, Biederman teaches reconfigurable logic in “multiple partitions” in the programmable fabric of an FPGA, adjustable in size. Ex-1005, ¶129, ¶59; ¶134 (“[C]oordinator circuit 110 may increase or decrease the amount of...*partition* available for instructions or programs^[11].”), Fig. 8:

¹¹ Biederman refers to FPGA logic designs as “FPGA programs.” *E.g.*, Ex-1005, ¶128.

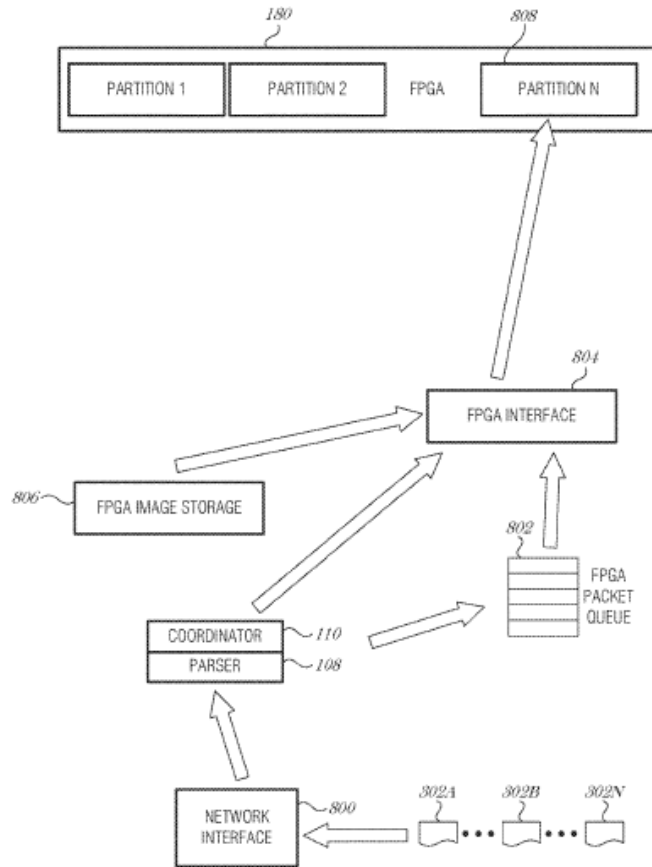


FIG. 8

Ex-1003, ¶141.

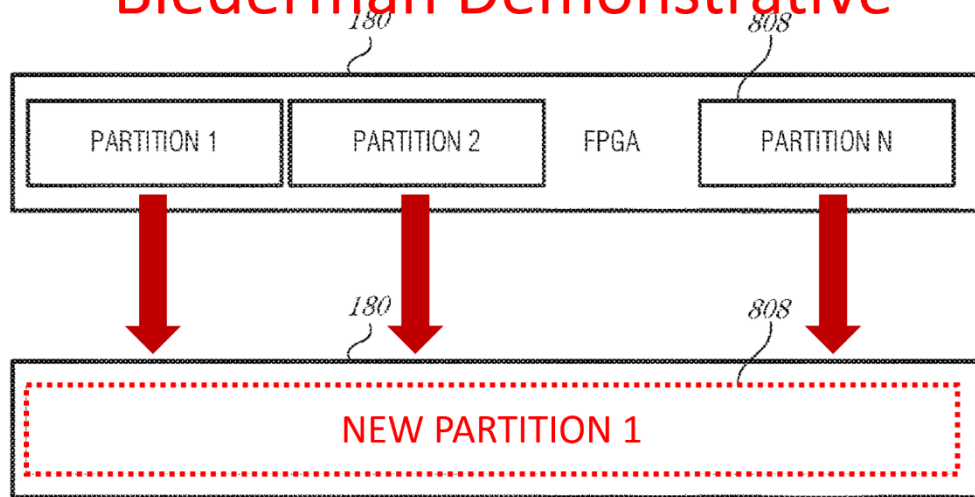
The coordinator circuit manages partitions, deciding which logic design to load to each partition and instructing the FPGA interface controller to load the designs. Ex-1005, ¶¶123-126, ¶128. Biederman adjusts the partition size to match a logic design that is chosen based on the priority of tasks assigned to the partition, choosing larger or smaller logic designs to meet the processing needs of a task. Ex-1005, ¶134. “Using a larger cache/partition space may act to reduce further cache/FPGA misses for complex processing, thereby ensuring faster processing

and a better user experience.” Ex-1005, ¶134. For example, “[h]igh-priority programs, such as a video conferences, may use complex processes to ensure real-time or near real-time audio and video presentation.” *Id.* Therefore, they would require more space (e.g., a larger partition with more programmable logic sectors) to load the user’s custom logic. Ex-1003, ¶142.

Biederman illustrates the operation of the coordinator circuit and FPGA partitions with an example of processing network packets. Ex-1005, ¶¶123-129, Fig. 8. Incoming packets are “parsed to determine a corresponding program to be run by the FPGA” and “[t]he packet parser 108 or coordinator circuit 110 determines the FPGA program needed to process [each of] the packet[s] 302A-N.” Ex-1005, ¶123. While incoming packets are processed, logic designs on partitions P1-PN are continuously adapted to the packet data being processed. Ex-1005, ¶¶130-131, Fig. 9. Logic designs that are not being used to process current packets are evicted and the space in the partition is reclaimed and reconfigured. Ex-1005, ¶129; Ex-1003, ¶143.

When smaller partitions are evicted to make room for a larger logic design (Ex-1005, ¶¶129-31), the smaller partitions are combined to form a single, larger sandbox partition. This occurs because multiple partitions divide up the limited amount of FPGA space for customer logic. Ex-1003, ¶144 (based on Figure 8):

Biederman Demonstrative



For example, when smaller logic in partitions 1 through N are replaced by larger video transcoding logic—e.g., when the next packet requires heavy processing for video transcoding—the smaller partitions are combined into a single, larger partition to accommodate resized partition 1. *See* Ex-1005, ¶134; Ex-1003, ¶144.

Motivation to Combine. Seshadri teaches an FPGA with programmable logic sectors partitioned into a shell area and a client-configurable circuit area, which can be further partitioned. Ex-1004, 2:50-3:64. Biederman teaches known techniques for partitioning an FPGA's programmable fabric and dynamically adjusting partition sizes. Ex-1005, ¶134. A POSITA would have been motivated to combine Seshadri's teachings with Biederman's teachings because it would have combined prior art elements according to known methods to yield the predictable result of adjusting partition sizes to fit the logic loaded on an FPGA partition. Ex-1003, ¶145.

Additionally, the combination provides several benefits. By sizing partitions to the logic to be hosted, it adds flexibility to the partitioning of logic sectors in the FPGA, allowing larger designs. This also increases FPGA utilization for cloud service providers, allowing more customers to be served with a smaller amount of hardware, and lowering cost. *See* Ex-1026, 000002-4; Ex-1003, ¶146.

A POSITA would have a reasonable expectation of success because the combination would not have changed the principle of operation for either reference. Seshadri's teachings would still be used to partition and reconfigure client-configurable circuits; Biederman's teachings would still be used to adjust the size of partitions. The operation of the programmable logic sectors, interconnects, and other FPGA elements would perform in the same way. Ex-1003, ¶147.

Seshadri and Biederman provide complementary teachings related to FPGA composition and operations. The combination adds minor partitioning details from Biederman. Configuring a partition out of previously-assigned logic sectors was straightforward because reprogramming portions of an FPGA's programmable fabric was known—reprogrammability was a central feature of FPGAs. A POSITA with 2 years of experience would have been fully capable of utilizing partitions on an FPGA. *Supra* §III; Ex-1003, ¶148.

18. ...

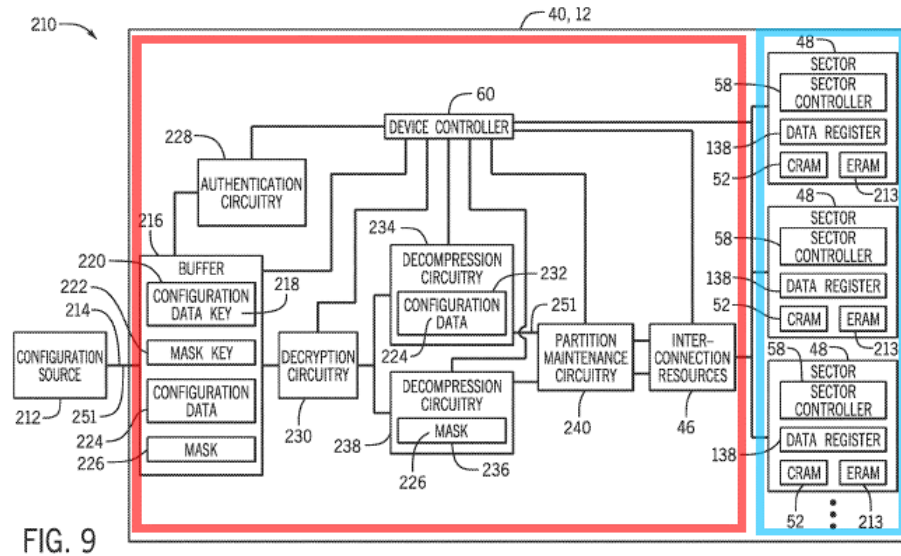
Claim 18 recites overlapping limitations with claim 9, which are satisfied for the same reasons. *Supra* §V.A.12[9] (incorporated by reference). Claim 18 recites a “single,” larger sandbox; this was explained above. *Id.*

B. Grounds 3-4

The Challenged Claims are obvious based on Atsatt-113. The combination with Biederman provides additional details for data flow architectures and claims 9/18. Ex-1003, ¶¶149-259. A POSITA would have been motivated to combine because both are Intel references that use FPGAs to accelerate multi-user VM processing. *Infra* §V.B.2[1pre], §V.B.3[10c]-[10d], §V.B.9 (subsections explaining motivation to combine); Ex-1003, ¶150.

**1. U.S. Patent Application Publication 2019/0095113A1
 (“Atsatt-113”)**

Atsatt-113 is §102(a)(2) prior art, filed September 28, 2017. Atsatt-113 “maintain[s] reconfigurable partitions in...a programmable device such as an FPGA,” assigned to different users. Ex-1006, ¶5, ¶27. The FPGA has a control area (left) and programmable logic sectors (right). Ex-1006, ¶¶30-31.



Atsatt-113 performs key-based authentication to unlock access to FPGA partitions.

Ex-1006, ¶¶41-42, ¶¶59-69; Ex-1003, ¶¶151-154.

Atsatt-113 is analogous art in the same field of endeavor and reasonably pertinent to the '943 patent (*supra* §III) because it configures PLDs for user-defined data processing in “a server having multiple virtual machines” and teaches a solution for reconfiguring FPGA partitions. Ex-1006, ¶5, ¶1, Abs; Ex-1003, ¶155.

2. Claim 1

[1pre]. A programmable device for data flow processing in a user-configurable server with swappable pods or cards, comprising:

Grounds 3-4 teach the preamble. Ex-1003, ¶¶156-172.

Atsatt-113 teaches the preamble.

Atsatt-113 teaches “programmable devices, such as field programmable gate arrays (FPGAs)” (Ex-1006, ¶1) which “contain programmable logic (e.g., logic blocks)...” Ex-1006, ¶3; Ex-1003, ¶157.

Atsatt-113 uses PLDs for data flow processing in servers, with multiple VMs, including data-center and network-routing applications. Ex-1006, ¶5, ¶3. Additionally, the incorporated How reference¹² discloses FPGA systems for, e.g., “video processing, digital signal processing, or any other application where the advantage of using programmable or reprogrammable logic is desirable.” Ex-1031, ¶58. Therefore, the combined teachings of Atsatt-113 and its incorporated reference confirm the known and obvious use of FPGAs for data-flow processing. *See id*; Ex-1003, ¶¶158-159.

Additionally, Atsatt-113 obviously teaches the transfer and processing of data as it flows through a data center because Atsatt-113 teaches using FPGAs in servers and “data centers.” Ex-1006, ¶3, ¶5. Such servers were often used for data flow processing in cloud computing, with FPGAs for accelerating commonly

¹² U.S. Patent Pub. No. 2016/0049941A1 (“How”) was incorporated by reference in Atsatt-113. Ex-1006, ¶29.

intensive data flows including multimedia data, AI training, etc. *Supra* §I; Ex-1003, ¶160.

Atsatt-113 teaches a user-configurable server with swappable cards. Ex-1006, ¶5. Users configured the server by loading custom logic onto the FPGA. *Infra* §V.B.3[1c]. Computers/servers typically included PCIe slots for inserting peripheral cards including FPGAs, and in data centers, VMs were commonly implemented using racks of servers with FPGA cards. *Supra* §I. Additionally, this would have been obvious given Atsatt-113's teachings of data centers and FPGAs in servers. Ex-1006, ¶3, ¶5; Ex-1003, ¶161.

Biederman teaches the preamble.

Biederman provides additional teachings for data-flow processing with swappable cards. Biederman uses FPGAs on “network interface card[s]” (NICs) in servers. Ex-1005, ¶25, ¶2, ¶¶28-29, Fig. 1. Biederman teaches multiple swappable PCIe cards on user-configurable network servers (Ex-1005, ¶¶174-176, ¶¶50-54, Fig. 1) including FPGAs the user configures by loading custom logic. *Infra* §V.B.3[1c]; Ex-1003, ¶¶163-165.

Biederman teaches using FPGAs for data-flow processing, including for “packet processing” of network data (Ex-1005, ¶2, ¶31) and video data (Ex-1005,

¶134, ¶152).¹³ Biederman teaches using FPGA cards to process AI image data, with data flowing through a data center processed over a given path. Ex-1005, ¶182-184; Ex-1003, ¶¶166-167.

Motivation to Combine. A POSITA would have been motivated to apply Biederman’s teachings to Atsatt-113. The references themselves suggest the combination: Atsatt-113 teaches FPGAs and servers (Ex-1006, ¶3, ¶5), while Biederman uses FPGA cards in servers for data-flow processing. Ex-1005, ¶¶174-176, ¶2, ¶¶182-184. A POSITA would have been motivated to apply Biederman’s well-known examples of how FPGA-equipped servers were used. This would have been a combination of prior art elements according to known methods to yield the predictable results of using Atsatt-113’s FPGA teachings to provide data-flow services in servers. Ex-1003, ¶168.

Both references are directed techniques for using FPGAs in servers. Since Atsatt-113 teaches techniques without detailing particular use cases, a POSITA would naturally look to known uses of FPGAs in servers, including Biederman’s teachings. Ex-1006, ¶¶3-5; Ex-1005, ¶2, ¶¶174-184. Atsatt-113 provides motivation by incorporating How (Ex-1006, ¶29), which applies Atsatt-113’s

¹³ The ’943 patent describes “video processing” as an example of data flow processing. Ex-1001, 9:51-10:31, 2:27-53, 7:45-46, Figs. 8, 11.

sectorized FPGA architecture to similar applications as Biederman. Ex-1031, ¶58; Ex-1003, ¶169.

The combination provides several benefits. Using Atsatt-113's reconfigurable FPGAs would have increased platform flexibility for common workloads. Biederman's applications often used custom processing, which Atsatt-113 provides on an FPGA. Ex-1006, ¶36, ¶5. The combination would therefore accelerate processing with lower energy/thermal requirements. *Supra* §I; Ex-1003, ¶170.

A POSITA would have had a reasonable expectation of success because Biederman simply provides uses for Atsatt-113's FPGA teachings. Thus, Atsatt-113's teachings would have been used in the same way Atsatt-113 already teaches. Ex-1005, ¶¶174-184. Furthermore, the references themselves suggest the combination, and a POSITA would have been more than capable of combining the teachings of these two references. *See supra* §III; Ex-1003, ¶171.

[1a] a programmable logic device (PLD) having a first region and a second region;

Atsatt-113 teaches [1a]. Ex-1003, ¶¶173-177. Figure 9 teaches a “programmable logic device (e.g., FPGA 40)” (Ex-1006, ¶59, ¶29) with a first control area region (annotated red) and a second region of programmable logic sectors (annotated blue):

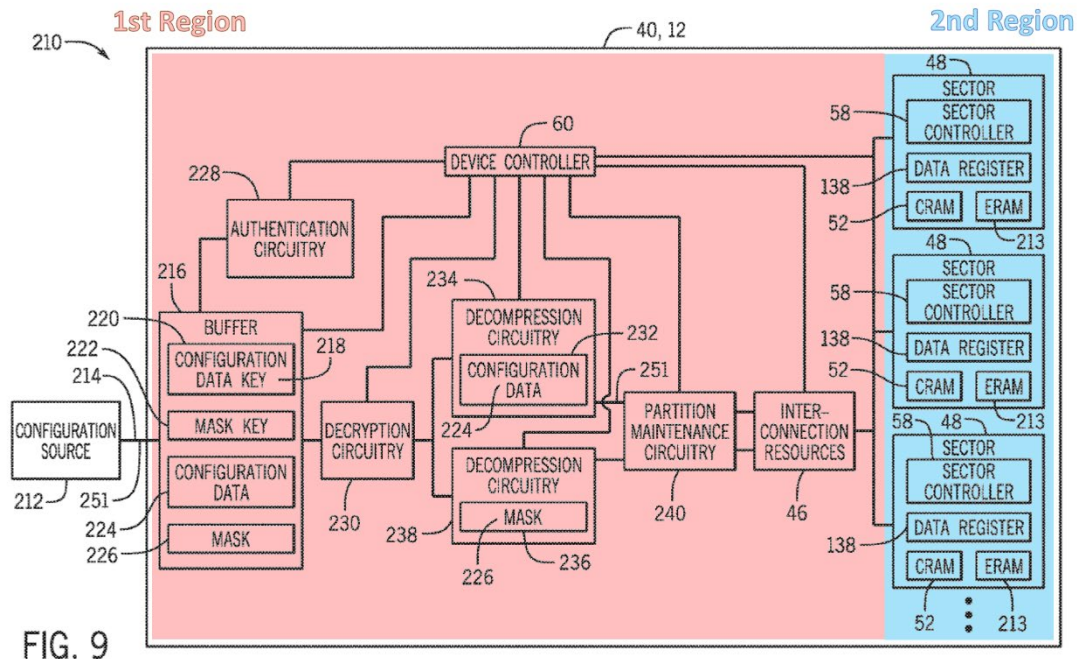


FIG. 9

Ex-1003, ¶173.

Atsatt-113’s PLD is an IC where the user can alter the connection pattern of the circuits. *Supra* §IV.A (claim construction). Atsatt-113 teaches an FPGA as one type of PLD with “integrated circuits” and logic “programmed (e.g., configured) after manufacturing.” Ex-1006, ¶3. When the user configures/reconfigures the FPGA logic circuits to perform a particular function, it alters the interconnection pattern of the logic circuits to implement a logic function provided by the user. *Id.* These teachings are consistent with the conventional use of FPGAs. *Supra* §I; Ex-1003, ¶174.

Atsatt-113 explains its teachings with an example “sectorized FPGA,” with the user-configurable logic areas divided into “any suitable number of programmable logic sectors.” Ex-1006, ¶29, ¶31, ¶38.

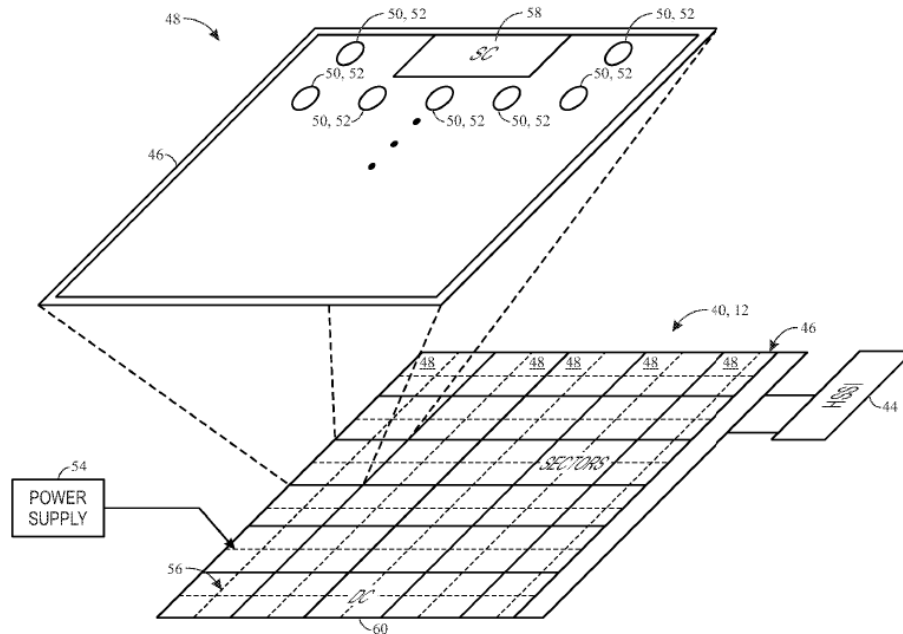


FIG. 2

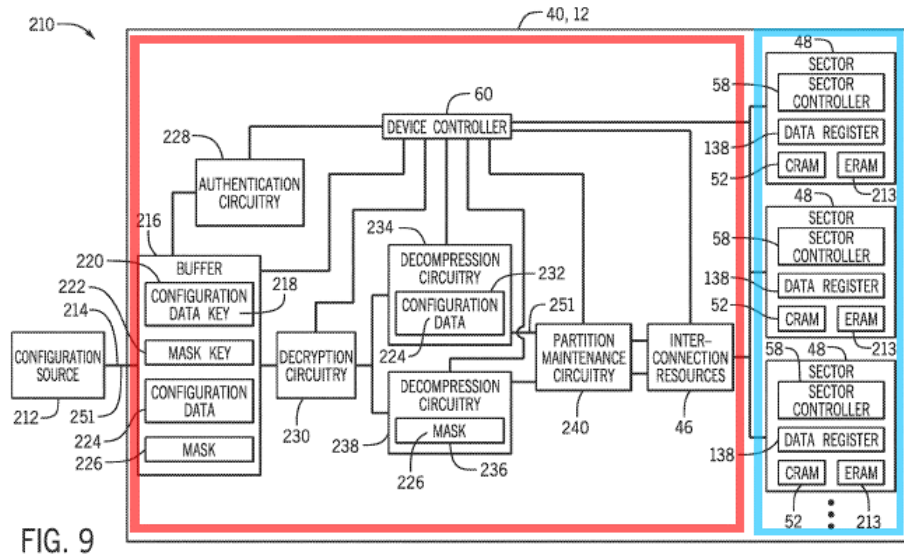
Ex-1003, ¶175.

[1b] the first region comprising a hardware-based or firmware-based router with a port enabling communication between the router and the second region; and

Atsatt teaches [1b]. Ex-1003, ¶¶178-187. Atsatt-113 teaches the first region (e.g., control area) comprising a router with a port.¹⁴ For example, the control area comprises functionality (e.g., in global resources, including interconnection resources and the device controller) to “route” data signals through the FPGA using interconnection resources (Ex-1006, ¶30), where the “signals may be transmitted as communication packets.” *Id.*, ¶33, Fig. 9. Atsatt-113 refers to

¹⁴ *Supra* n. 6, §II.C (PO characterization of router/port).

FPGA “resources that are shared between...and outside of the physical regions of the [programmable logic] sectors” as “global resources,” which includes interconnection resources. Ex-1006, ¶39 (“spanning resources that span physical partitions and/or sectors”), ¶55-56:



The global resources include device controller 60 which routes communications from outside the FPGA to the destination sector. Ex-1006, ¶33. For example, external communication to/from the HSSI interface are routed over pathways provided by the interconnection resources. Ex-1006, ¶33:

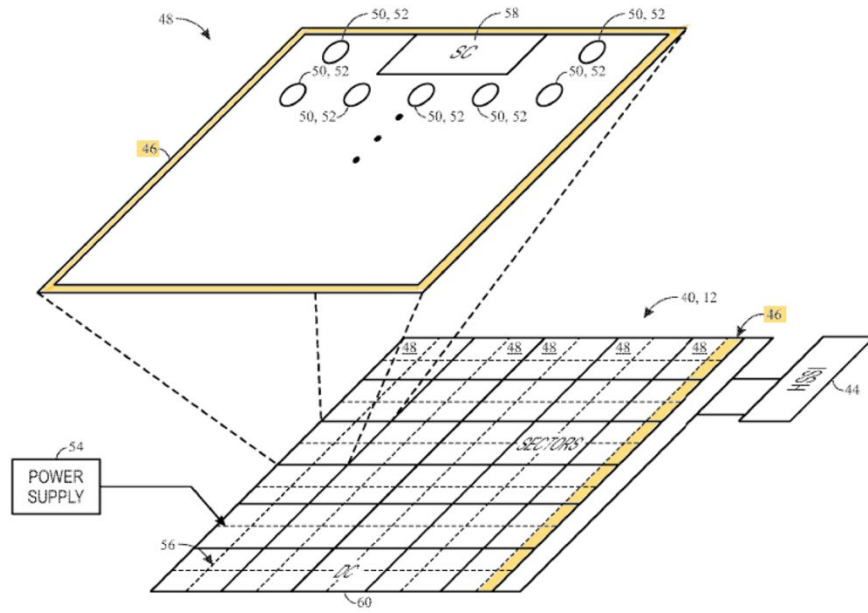


FIG. 2

Communications with external devices travel between the router functionality in global resources and the programmable logic sectors because the router functionality acts as an intermediary for receiving inbound communications and sending outbound communications to the host or other external devices. Ex-1006, ¶¶30-33, ¶39 (discussing external PCIe communication), ¶55 (HSSI), Fig. 2; Ex-1003, ¶¶179-181, ¶185.

Additionally, when the user's circuit in a partition is updated, channel information is used to route configuration data to the programmable logic sectors by “inform[ing] the interconnection resources 46 where to send” it. Ex-1006, ¶71, ¶100, Fig. 10; Ex-1003, ¶¶183-184.

Therefore, Atsatt-113 includes router functionality in the global resources to route communications to the second region, which further includes port functionality to enable communication between the global resources and the second region. For example, the interconnection resources include functionality serving as a port for communication between circuits in the control area and the programmable logic sectors. Ex-1006, ¶¶59-66. Commands and data from the first region are forwarded to their destination, such as the configuration RAM (“CRAM”) in programmable logic sectors, as packets. *Id.* Atsatt-113 provides an example where configuration data is routed from the device controller in the control area: “validated or verified configuration data 224 is sent to the interconnection resources 46 to be distributed to one or more sectors 48 of the FPGA 40.” Ex-1006, ¶¶81. Atsatt-113 further describes interconnection resources as including ports for input/output of communication between different blocks of the FPGA. Ex-1006, ¶¶55, ¶¶39. Figure 9 shows the logical connections established through the port from the device controller and interconnection resources to the programmable logic sectors:

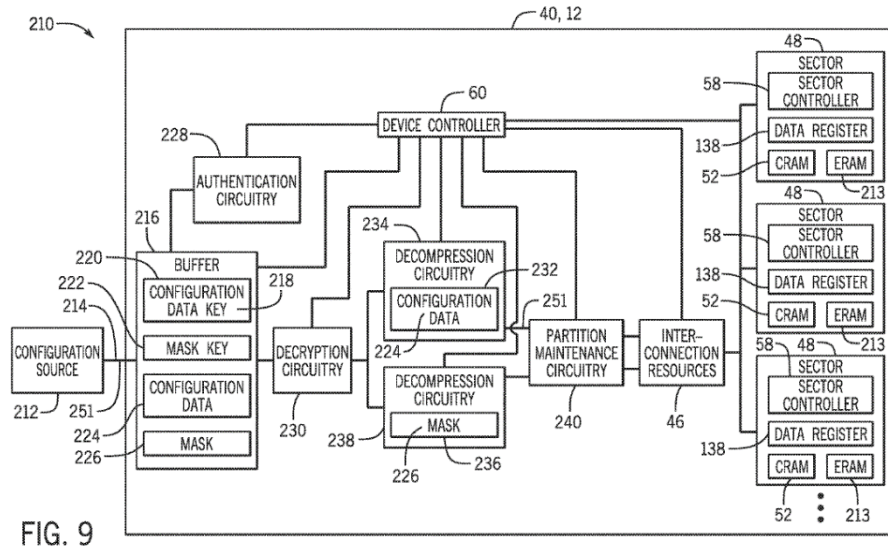


FIG. 9

Ex-1006, Fig. 3; Ex-1003, ¶¶182-183.

Atsatt-113’s router functionality is hardware-based because the above-discussed teachings are implemented on the hardware-programmable fabric of the FPGA. Ex-1006, ¶39 (“The *programmable fabric 100* also includes *global resources 112, 114, 116, and 118....*”); Ex-1003, ¶186.

[1c] the second region comprising one or more sandboxes with user-definable programmable electronic circuits of the PLD,

Atsatt-113 teaches [1c]. Ex-1003, ¶¶188-195. The programmable logic sectors comprises multiple sandboxes, e.g., “logical grouping[s] of logical elements within the programmable fabric” of the FPGA.¹⁵ Ex-1006, ¶36. Atsatt-113 groups sectors into partitions for “different [] users” to load dedicated logic,

¹⁵ In the ’943 patent’s embodiments, sandboxes are designated areas of an FPGA. Ex-1001, 5:19-49.

performing “different functions,” into their particular partitions. Ex-1006, ¶40; Ex-1003, ¶189. Figure 3 illustrates the allocation of “partitions” across sectors. Ex-1006, ¶40.

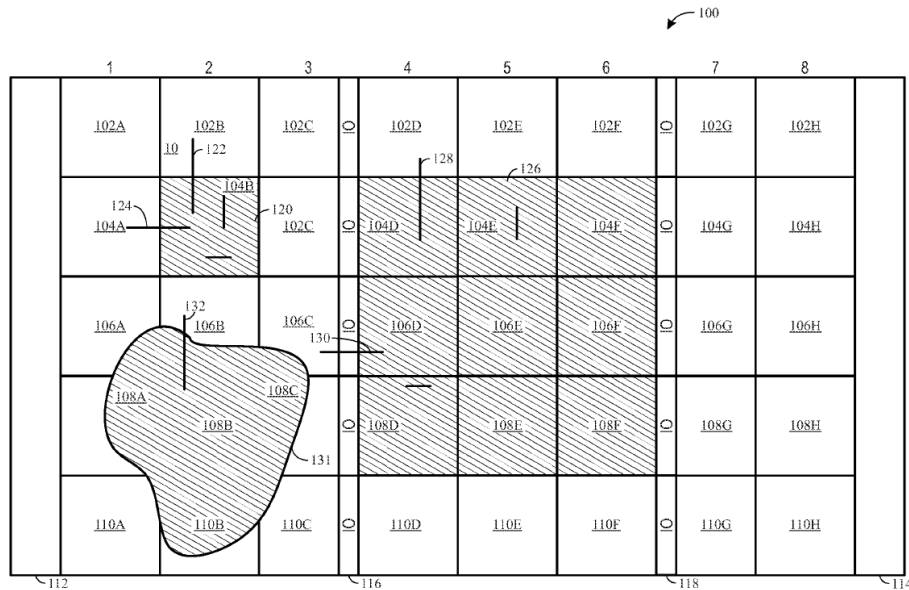


FIG. 3

Atsatt-113 teaches that the partitions are sandboxes where the partitions and related resources are isolated from partitions assigned to other users “to ensure they are not modified by partitions that do not own them.” Ex-1006, ¶41. Atsatt-113 teaches various mechanisms for isolating partitions: for example, maintaining an “authenticated list” of addresses to control access to the embedded RAM associated with a sector. Ex-1006, ¶54. Atsatt-113 also explains how a “violation of resource separation” can be prevented during configuration of user partitions (*id.*, ¶53) and uses masks to enforce isolation by defining the contours of each

sandbox (e.g., the user partitions) and the resources a partition can access. *Id.*, ¶¶41-42; Ex-1003, ¶¶190-192.

The partitions comprise user-definable programmable electronic circuits of the PLD; they are portions of the FPGA’s programmable fabric. Ex-1006, ¶40, ¶36. Atsatt-113 explains its FPGA contains physical “circuits” and “include[s] logic that may be programmed (e.g., configured) after manufacturing[.]” *Id.* ¶3. Logic loaded to the user partitions is reconfigurable by the user and therefore user-definable. Ex-1006, ¶5 (“[E]nable a user to change a region allocated to the user....”), ¶36, ¶59 (“configuration data 224 (e.g., *used to implement a user design on the partially reconfigurable region*).”). ICs such as FPGAs comprise electronic circuits. Ex-1006, ¶3, ¶5; Ex-1003, ¶193.

[1d] wherein the port comprises: a bridge being lockable to prevent user access, and unlockable to enable user access, to each of the sandboxes and the user-definable programmable electronic circuits therein.

Atsatt-113 teaches [1d]. Ex-1003, ¶¶195-201. The port includes interconnection resources (*supra* §V.B.2[1b]), which further comprise bridge functionality that is lockable/unlockable¹⁶ to prevent (by locking) or enable (by unlocking) user access to configure each partition. Ex-1006, ¶¶66-81. First, “an authentication key [is] used to authenticate the mask” used for

¹⁶ *Supra* n. 8, §II.C (explaining PO’s allegations).

configuration/reconfiguration of the FPGA. Ex-1006, ¶59. The device controller authenticates “the mask 226 with the mask authentication key 222.” Ex-1006, ¶62, ¶¶61-62. This is then applied to lock/unlock access to different resources and programmable logic sectors based on the “mask for a partition that indicates resources to which the partition is associated.” Ex-1006, ¶5, ¶41-42, ¶56 (the masks “describe this ownership” of resources). In short, the mask mechanism is used to enable or prevent user access to each partition (comprised of specific sectors). Ex-1003, ¶196.

Figure 8 shows the flow of mask generation and use, the masks setting ownership for global resources and user partitions. Ex-1006, ¶¶55-56, Fig. 8. When user partitions are reconfigured, the appropriate masks are applied to unlock the appropriate partition while leaving other partitions locked. Ex-1006, ¶56, Fig. 8 (step 202); Ex-1003, ¶197.

For embodiments that use a mask,¹⁷ “the mask 226 may also be sent to the interconnection resources” to unlock the partitions where the user has access,

¹⁷ A POSITA would have been motivated to use masks as taught by Atsatt-113 because they increase the security and simplicity of authenticating and managing users’ partition permissions while preventing users from altering partitions they do not have access to. Ex-1006, ¶26; Ex-1003, ¶198.

allowing FPGA configuration data to be sent to those partitions, while leaving other partitions locked. Ex-1006, ¶81, ¶30 (“Each programmable logic sector 48 may include a number of programmable logic elements 50 having operations defined by configuration memory 52 (e.g., configuration random access memory (CRAM)).”), ¶34, ¶¶43-45, ¶54. The interconnection resources are responsible for sending configuration data to the programmable logic sectors. Ex-1006, ¶71, ¶100, Fig. 10; Ex-1003, ¶198.

Therefore, functionality in the interconnection resources acts as a bridge for accessing specific programmable logic sectors and associated resources. When these transactions pass over the interconnection resources, they are modified based on the applicable mask state. Ex-1006, ¶41. This mechanism locks the interconnection resources for the associated user to prevent modification of partitions when the user’s key has not been authenticated; the interconnection resources are unlocked when the key is authenticated. *Id.*, ¶81, ¶26, ¶59. Atsatt-113 teaches sending the mask to the interconnection resources to enforce the mask when distributing configuration data to partitions. *See id.* A POSITA would have understood this and found it obvious from Atsatt-113’s teachings because the interconnection resources distribute configuration data to partitions, and the mask is authenticated to confirm whether the user has access to specific partitions. *Id.*,

¶81, ¶71, ¶51, ¶57, ¶¶66-67. That is why Atsatt-113 sends the mask to the interconnection resources—to enforce the mask. *Id.*, ¶81; Ex-1003, ¶199.

The FPGA partitions comprise user-definable programmable electronic circuits. *Supra* §V.B.3[1c]; Ex-1003, ¶200.

3. Claim 10

10[pre]-[a] ...

[10pre]-[10a] are recited in [1pre]-[1a]. They are satisfied for the same reasons. *Supra* §V.B.2[1pre]-[1a] (incorporated by reference); Ex-1003, ¶203.

[10b] ...

[10b] overlaps with [1b]. For both claims, the first region comprises a router with a bridge and port.¹⁸ Ex-1003, ¶204.

As explained above, Atsatt-113 teaches the first region comprising a hardware-based router with a bridge and port that enables communication between the router and programmable logic sectors. Ex-1006, ¶¶30-33, ¶¶55-66, ¶39; *supra* §§V.B.2[1b], [1d] (incorporated by reference); Ex-1003, ¶205.

[10c] the second region comprising a plurality of sandboxes...

[10c] recites overlapping language with [1c] that is satisfied for the same reasons. *Supra* §V.B.2[1c] (incorporated by reference). [10c] additionally

¹⁸ The '943 patent uses router and bridge interchangeably. *Supra* n. 9

requires a plurality of sandboxes, which is taught by Grounds 3-4. Ex-1003, ¶¶206-210.

Ground 3

Atsatt-113 explains its PLD includes a plurality of “partitions” for different users. Ex-1006, ¶40; Ex-1003, ¶¶207; *supra* §V.B.2[1c].

Ground 4.

Biederman teaches a plurality of sandboxes. *Supra* §V.A.2[10c] (incorporated by reference); Ex-1003, ¶209.

Motivation to Combine. Atsatt-113 provides express motivation by teaching partitions (plural) for different users. Ex-1006, ¶40. Biederman provides mechanisms for managing and resizing multiple partitions on an FPGA. Ex-1005, ¶134. A POSITA would have been motivated to combine these teachings to use multiple partitions and adjust their size. *Infra* §V.B.9 (subsection on motivation to combine, incorporated by reference); Ex-1003, ¶210.

[10d] an internal communication network, coupling the port of the bridge of the router and the one or more sandboxes for the data flow processing,
--

Grounds 3-4 teach [10d]. Ex-1003, ¶¶211-220.

Ground 3

Atsatt-113's global resources includes functionality for a router with a port. Ex-1006, ¶55, ¶39, Fig. 9; *supra* §V.B.2[1b] (incorporated by reference). Atsatt-113's interconnection resources includes functionality to “act as a network” that couples the port to the partitions in the programmable logic sectors. Ex-1006, ¶33. This internal communication network couples global resources to the sector controllers for the programmable logic sectors comprising each user partition. Ex-1006, Fig. 2.

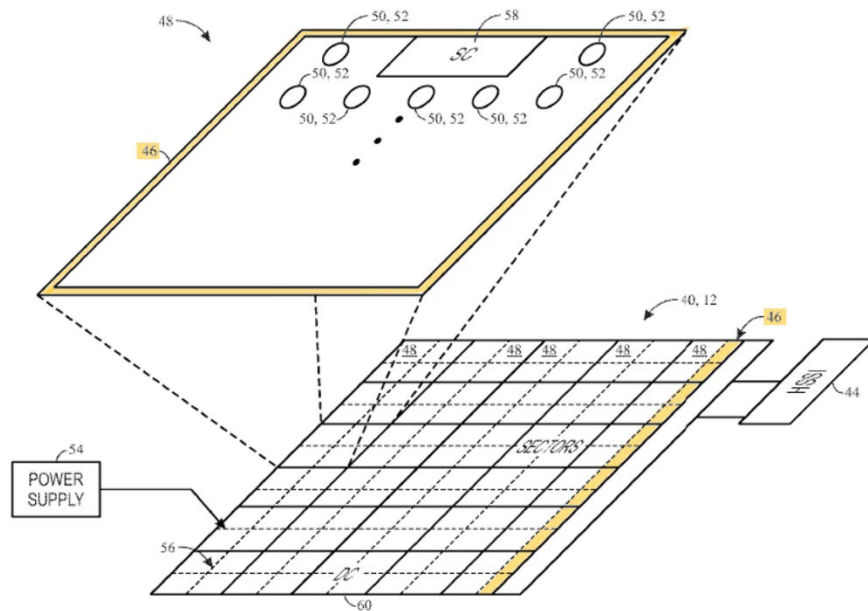
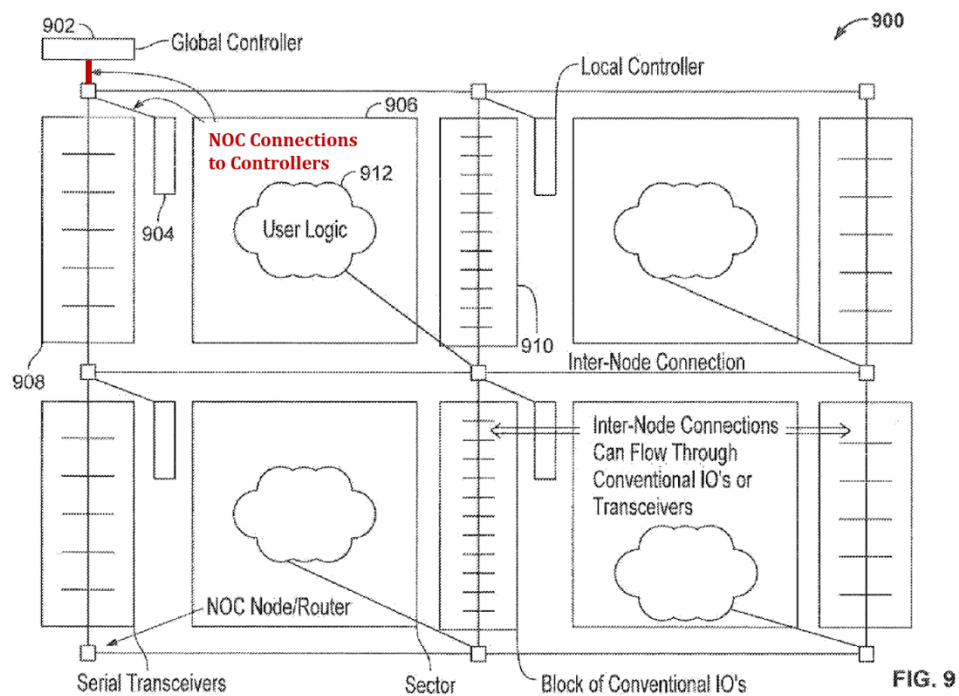


FIG. 2

Ex-1003, ¶¶212-213.

Atsatt-113 teaches further details through the incorporated How reference:¹⁹



¹⁹ Atsatt-113 incorporates How’s disclosures regarding the structure, configuration, and operation of “a sectorized FPGA of the type described in” How. Ex-1006, ¶29.

Therefore, the “NOC Connection” to the global controller enables communication between the FPGA global resources and the programmable logic sectors. *See id.*; Ex-1003, ¶214.

Taking Atsatt-113’s and How’s teachings together, Atsatt-113’s interconnection resources includes a port (including the NOC Connection) and a network-on-chip coupling the port to the programmable logic sectors, as taught by How. Ex-1033, ¶39, ¶21, Fig. 9 A POSITA would also have found this obvious because Atsatt-113 explains how interconnection resources “may act as a network” (Ex-1006, ¶33), while How provides details regarding that network (Ex-1031, ¶39). Ex-1003, ¶216.

Atsatt-113 states its sectorized FPGA comes from How. Ex-1006, ¶29. Both share an assignee (Altera), overall architecture, and a co-inventor. Atsatt-113’s control area corresponds to How’s global controller because both control the operations of the FPGA. Ex-1006, ¶33, ¶¶61-66; Ex-1031, ¶¶3-4, ¶¶25-29, ¶¶32-37. How’s NoC falls within the scope of the interconnection resources because it interconnects components throughout the FPGA. Ex-1031, ¶¶37-38 (“network of *connections* between the global controller and each sector’s local controller.”), ¶39; Ex-1003, ¶215.

Ground 4

Biederman provides additional teachings for using FPGA partitions for data flow processing, including “packet processing,” image data, and AI data. Ex-1005, ¶2, ¶¶182-184; *supra* §V.B.2[1pre] (incorporated by reference); Ex-1003, ¶218.

Motivation to Combine. The same analysis for claim 1 applies here. *Supra* §V.B.2[1pre] (motivation to combine, incorporated by reference). Ex-1003, ¶219.

[10e] ...

[10e] recites overlapping language with [1d] that is satisfied for the same reasons. *Supra* §V.B.2[1d] (incorporated by reference); Ex-1003, ¶¶221-223.

Grounds 3-4 further teach a plurality of sandboxes. *Supra* §V.B.3[10c]-[10d] (incorporated by reference). Because Atsatt-113 teaches multiple partitions, its teachings for locking/unlocking the bridge apply to a plurality of sandboxes. Ex-1003, ¶222.

4. Claim 2

Grounds 3-4 teach claim 2. Ex-1003, ¶¶224-234.

Atsatt-113 teaches the router (e.g., functionality in global resources) utilizing various protocols to communicate with external clients (using HSSI) or with the host (using PCIe), which is routed to the appropriate sandbox. Ex-1006, ¶39, ¶55. Therefore, the routing functionality in global resources translates

between a first external protocol (e.g., PCIe/HSSI) and a second protocol (FPGA's internal protocol); Ex-1003, ¶225.

External protocols such as PCIe and HSSI were used for communication between ICs and were optimized for circuit boards and cables, often with serial signals. Ex-1033, 0000017, 000029-32; Ex-1032, 000002, 000011-13. The incorporated²⁰ How reference discloses other communication protocols. Ex-1031, ¶21 (“general purpose I/Os,” “DDR memory interfaces”), Fig. 1. In contrast to PCIe and HSSI, communication within FPGAs was optimized for internal, often parallel buses, including for example interconnect resources between different parts of the FPGA. Atsatt-113 teaches an internal protocol, different from the external protocols, because the FPGA interfaces with a variety of protocols that require translation for interoperability. *See* Ex-1031, ¶36, Fig. 7; Ex-1003, ¶¶226-227.

Atsatt-113 teaches the router performing these translations for messaging with system resources external to the FPGA. Atsatt-113 teaches FPGAs in networked servers (Ex-1006, ¶5, ¶3), which typically used memory elsewhere in the server or remote storage because FPGAs had limited memory capacity. Ex-1003, ¶¶228-229.

²⁰ *Supra* n. 19.

Biederman further teaches examples of external resources, including memory, storage devices, other FPGAs, processors, co-processors, etc. Ex-1005, ¶¶53-54, ¶59, ¶¶193-196, Fig. 1, Fig. 18. The NIC, which includes the FPGA, “may query or otherwise interface with various components of the system” including those “external from the system[.]” Ex-1005, ¶198; Ex-1003, ¶¶230-231.

5. Claim 11

The overlapping portions of claim 11 are satisfied for the same reasons explained for claim 2. *Supra* §V.B.4. Atsatt-113 further teaches the first protocol (e.g., HSSI or PCIe) is used external to the second region (FPGA programmable logic sectors) and the second protocol is used in the internal communication network in the interconnection resources of the PLD/FPGA. Ex-1006, ¶39, ¶55, Fig. 2; *supra* §V.B.4; §V.B.3[10d] (explaining internal network, incorporated by reference); Ex-1003, ¶¶233-234.

6. Claims 3/12

Atsatt-113 teaches claims 3 and 12. Ex-1003, ¶¶235-240. Each of Atsatt-113’s user partitions comprises user-programmable digital circuitry (e.g., programmable logic sectors of the FPGA). Ex-1006, ¶3. The user partitions (sandboxes) comprise user-definable programmable electronic circuits. *Supra* §V.B.2[1c] (incorporated by reference). The digital circuitry for each partition is

programmable by each owner-user (Ex-1006, ¶5), with each partition being separately programmed with configuration data for that partition. Ex-1006, ¶59; Ex-1003, ¶¶236-237.

Each programmable logic sector comprises digital circuitry because each is a collection of digital logic elements. Ex-1006, ¶30 (“combinational or sequential logic circuitry...look-up tables”). Atsatt-113’s teachings refer to digital circuitry using digital signals and discrete output. To the extent PO argues this circuitry is not digital, it would satisfy the claims as analog circuitry. Ex-1003, ¶238.

Regarding claim 12, Grounds 3-4 teach a plurality of sandboxes. *Supra* §V.B.3[10c] (incorporated by reference); Ex-1003, ¶239.

7. Claims 4/13

Atsatt-113 teaches claims 4 and 13. Ex-1003, ¶¶241-245. Atsatt-113’s port comprises a bridge that is lockable to prevent reconfiguration of the bridge and router functionality in the control area while enabling reconfiguration of the programmable logic sector partitions. Atsatt-113 teaches the global resources—including routing and bridge functionality—being assigned to a base partition that is “configured once and not updated by partition updates” such as reconfiguration of the user partitions. Ex-1006, ¶55.

Because the base partition is not owned by any user, it is locked by setting a “not allowed mask[]” for other user partitions (Ex-1006, ¶56), using Atsatt-113’s

mask mechanism for locking/unlocking access to partitions. *Supra* §V.B.2[1d] (incorporated by reference). The base partition is locked after initial setup; therefore, it is locked when user partitions are unlocked to enable reconfiguration of the sandboxes. Ex-1006, ¶55; Ex-1003, ¶¶242-243.

Regarding claim 13, Grounds 3-4 teach a plurality of sandboxes (*supra* §V.B.2[10c] (incorporated by reference)), and the above teachings apply equally to multiple partitions. Ex-1003, ¶244.

8. Claims 5/14

Atsatt-113 teaches claims 5 and 14. Ex-1003, ¶¶246-253. Atsatt-113 discloses a security layer (e.g., device controller 60 and authentication circuitry or engine 228) in the port and bridge functionality of the global resources that authenticates configuration data and masks to verify permission to reconfigure the partitions. Ex-1006, ¶62 (“The device controller 60 may use authentication circuitry or engine 228...to authenticate...the mask 226 with the mask authentication key 222.”); ¶62 (authenticating using “a secure hash algorithm (e.g., SHA-1).”). Figure 16 (steps 382-398) shows the authentication flow up to reconfiguring the designated user partition. Ex-1006, ¶¶93-96. While Atsatt-113 refers to the device controller performing these steps, the teachings apply equally to authentication circuitry 228 because Atsatt-113 explains that the device controller uses the authentication circuitry to that end. Ex-1006, ¶62. Indeed,

Atsatt-113 states that this security layer functionality can “be implemented by the device controller 60 and/or one or more sector controllers 58 of the FPGA 40.”

Ex-1006, ¶93; Ex-1003, ¶¶247-250.

Functionality in the interconnection resources use this security functionality to enable or prevent access to a particular partition’s sectors because the authentication of the mask is part of the process for locking/unlocking access to programmable logic sectors. Ex-1006, ¶81, ¶¶41-42, ¶¶59-69; *supra* §V.B.2[1d] (incorporated by reference); Ex-1003, ¶251.

Regarding claim 13, Atsatt-113 teaches a plurality of sandboxes. *Supra* §V.B.2[10c] (incorporated by reference). The above teachings equally apply to multiple partitions. Ex-1003, ¶252.

9. Claims 9/18

Biederman teaches claims 9 and 18, as explained for Ground 2. That analysis applies here. *Supra* §V.A.12 (incorporated by reference); Ex-1003, ¶¶254-259.

Motivation to Combine. Biederman teaches a known technique to adjust the size of partitions on an FPGA. Ex-1005, ¶134. Atsatt-113 was ready for improvement because it teaches a known configuration of an FPGA with multiple partitions loaded with various circuits that differ in size. Ex-1006, Abstract, ¶¶36-38, ¶¶57-59, Fig. 3. A POSITA would have been motivated to combine Atsatt-

113's and Biederman's teachings because it would apply a known technique to a known device ready for improvement to yield the predictable result of adjusting partition sizes to fit the logic loaded on the partition. Ex-1003, ¶256.

The combination provides several benefits. Sizing partitions to logic adds flexibility to partitioning of logic sectors in the FPGA, which maximizes utilization and reduces cost. *See* Ex-1026, 000002-4 (discussing efficiency of sharing resources). Ex-1003, ¶257.

A POSITA would have had a reasonable expectation of success because the combination would not have changed the principle of operation for either reference. Atsatt-113's teachings would still be used to partition and reconfigure user partitions. Biederman's teachings would still be used to adjust the size of partitions. The operation of the programmable logic sectors, interconnects, and other elements of the FPGA in the combination would perform in the same way as they do in Atsatt-113 and in Biederman. Ex-1003, ¶258.

Atsatt-113 and Biederman provide complementary teachings related to the composition and operation of FPGAs. The combination adds minor details from Biederman that are compatible with Atsatt-113's teachings. A POSITA would have understood Biederman to teach the obvious result of Atsatt-113's teachings of reconfiguring partitions with different logic that may be larger than the previously

loaded logic. A POSITA with 2 years of experience would have been fully capable of making the combination. *Supra* §III; Ex-1003, ¶259.

C. Ground 5

Claims 6 and 15 are obvious based on each of Grounds 1-4 in view of Tarafdar. ¶¶260-288.

1. “Enabling Flexible Network FPGA Clusters in a Heterogeneous Cloud Data Center” (“Taraftdar”)

Taraftdar is §102(a)(1) prior art. Taraftdar was published in print by February 2017 in the proceedings for FPGA’17; it was publicly available because it was published for distribution at an industry conference, open to the interested public. Ex-1022, ¶9, ¶11. Taraftdar was also published on the ACM Digital Library online February 22, 2017; it was publicly available on the Internet through keyword search functionality of the ACM Digital library, which was commonly used by the interested public. *Id.*, ¶¶7-8, ¶10; Ex-1003, ¶¶261-263.

Taraftdar teaches the use of management/performance-analytics tools to orchestrate VMs on FPGAs in data centers. Ex-1007, 000002, Abs, 000007-9; Ex-1003, ¶264.

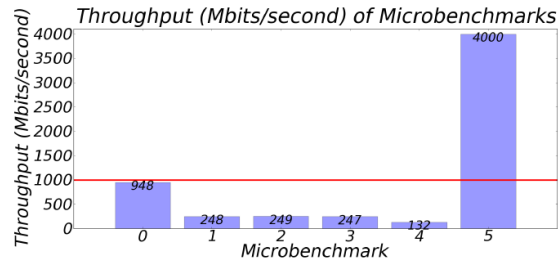


Figure 17: Throughput of the four microbenchmarks with two additional datapoints.

Tarafdar is analogous art in the same field and reasonably pertinent to the '943 patent (*supra* §III), because it optimizes and reconfigures FPGA-based clusters for data processing in heterogeneous data centers. Ex-1007, Abs., 000001; Ex-1003, ¶265.

2. Claim 6

6. The programmable device of claim 1, further comprising: the router configured to perform analytics on resources of the programmable device, on a per module basis, including determining communication bandwidth and resource utilization.

Tarafdar teaches claim 6. Ex-1003, ¶¶266-288. Tarafdar discloses “Cloud Cluster Management Tools” that perform analytics on FPGA resources to adjust the deployment of FPGA modules in cloud clusters. Ex-1007, 000002 (analyzing “metrics such as performance, *resource utilization*, and CPU usage.”). Tarafdar teaches known tools for “orchestration and load balancing using the aforementioned metrics.” *Id.* Tarafdar performs analytics by determining the

utilization of different resources and making decisions to adjust the configuration of the cluster based on these metrics.²¹ Ex-1003, ¶269.

The router (in the first region) is configured to perform these analytics. Like Seshadri and Atsatt-113, Tarafdar discloses a wrapper region, which Tarafdar calls a “hypervisor,” between the device interface (PCIe) and the user-configurable region of the FPGA, for routing and management functions. Ex-1007, 000003, Fig. 2:

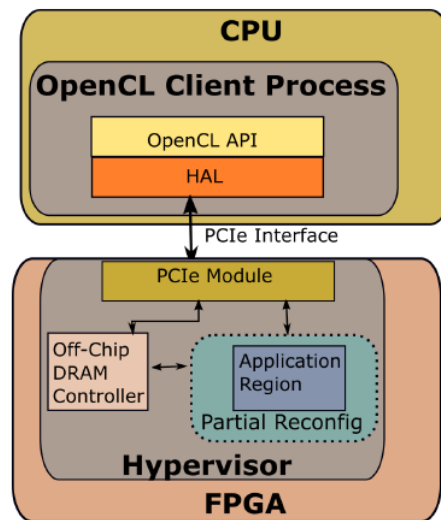


Figure 2: System diagram of the SDAccel platform.

Ex-1003, ¶270.

Tarafdar monitors and analyzes the resources at the router, in this Hypervisor region, because Tarafdar “monitor[s] and updat[es] our clusters using

²¹ The '943 patent confirms this falls within its scope. Ex-1001, 5:11-18; Ex-1003, ¶269.

FPGA metrics.” Ex-1007, 000002. Moreover, this would have been obvious because Tarafdar uses metrics to manage the FPGAs, and the Hypervisor region carries out the OpenCL API calls to manage the FPGAs. Ex-1007, 000003. This management region on the FPGA has the most direct and immediate access to overall FPGA metrics and resources, and Tarafdar teaches collecting “internal FPGA” metrics. Ex-1007, 000009. This would also reduce network overhead. A POSITA would have expected FPGAs to collect and provide such data because it was customary for prior-art ICs in similar applications. Ex-1003, ¶271.

Tarafdar teaches monitoring and analytics on a per-module basis for network throughput and resource overhead. Ex-1007, 000008-9. This was obvious because collecting these metrics for each module boosted efficiency and facilitated cluster orchestration by allowing modules to be assigned tasks based on their current status—one of Tarafdar’s stated goals. Ex-1007, 000001-2; Ex-1003, ¶272.

Tarafdar teaches determining communication bandwidth on a per-FPGA module basis, including network throughput determined for each microbenchmark in Figure 17, that are for user-configurable Application Regions. Ex-1007, 000008-9, 000005:

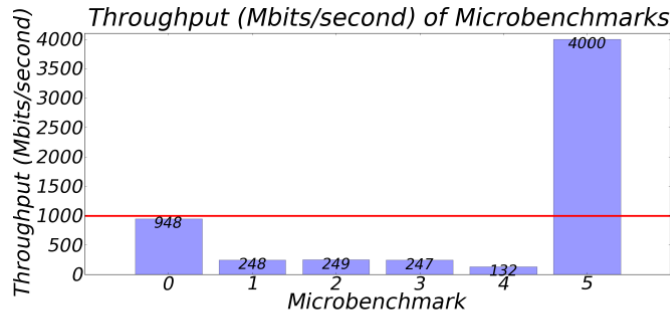


Figure 17: Throughput of the four microbenchmarks with two additional datapoints.

For example, Microbenchmarks 4 and 5 determine communication bandwidth for each FPGA. Ex-1007, 000009 (“throughput between two virtual machines,” “calculated by using the bus width”). Additionally, Microbenchmark 1 provides measurements for a single FPGA module between it and the host. Ex-1007, 000009. These microbenchmarks teach the measurement of communication bandwidth, and Tarafdar generally teaches and suggests monitoring and analyzing metrics on the FPGA. Therefore, Tarafdar suggests carrying out the determination of network throughput at the FPGA. Ex-1003, ¶¶273-277.

Tarafdar teaches determining resource utilization, including “resource overhead” using “absolute numbers” or “the percentage of the device total” for utilization of FPGA resources. Ex-1007, 000009; Ex-1003, ¶278.

Motivation to Combine. Seshadri, Atsatt-113, and Biederman teach PLDs/FPGAs in data centers. *Supra* §V.A.1-2, §V.B.1. A POSITA would have been motivated to apply Tarafdar’s related techniques for monitoring/analyzing resource utilization and bandwidth of FPGAs in cloud computing data centers. Ex-

1007, 000002, 000007-10. Applying these teachings to known FPGA architectures (Seshadri, Atsatt-113, and/or Biederman), would have yielded the predictable result of monitoring and analyzing resource utilization for each FPGA module. Thus, the combination of Tarafdar with each of Grounds 1-4 would have been a simple combination of prior art elements according to known methods to yield predictable results. Ex-1003, ¶280, ¶284.

A POSITA would have been motivated to apply Tarafdar's teachings to Grounds 1-4 to take advantage of several benefits. For example, Tarafdar "provide[s] an easy way" to monitor and orchestrate large FPGA clusters (Ex-1007, 000001-2. The use of such techniques "are helpful for getting reliable performance." *Id.* A POSITA would have recognized such benefits and would have been motivated to apply Tarafdar's teachings to distributed computing systems using FPGAs, such as those of Seshadri, Atsatt-113, and Biederman, in order to better administrate the nodes in the cloud and to increase FPGA usage efficiency. *See id.*; Ex-1003, ¶281, ¶285.

Additionally for Grounds 1-2, Seshadri suggests "various modifications and changes" (Ex-1004, 29:62-30:11); a POSITA would have been motivated to apply Tarafdar's teachings in the same spirit. Ex-1003, ¶280.

Additionally for Grounds 3-4, a POSITA would have been motivated to apply Tarafdar's teachings to a server with multiple PCIe cards because it

optimizes utilization between the FPGAs. Ex-1007, 000002, 000004, Fig. 4; Ex-1005, ¶¶52-54, Fig. 1; Ex-1006, ¶39, ¶55, Fig. 2.; Ex-1003, ¶286.

A POSITA would have had a reasonable expectation of success in the combinations. The combinations would not have changed the principle of operation of any of the references because it simply applies Tarafdar's FPGA-monitoring and analyzing teachings to Seshadri, Atsatt-113, and either in combination with Biederman, which likewise teach FPGAs. Therefore, the combinations simply use each reference for its intended purpose. Ex-1003, ¶282, ¶287.

A POSITA would have found the combinations obvious and would have had a reasonable expectation of success in doing so, because the combinations are relatively simple. Tarafdar demonstrates its teachings applied to an FPGA, and a POSITA would have been fully capable and understood how to apply it to other FPGAs. *Supra* §III; Ex-1003, ¶283, ¶288.

3. Claim 15

Claim 15 is nearly identical to claim 6. Tarafdar teaches the overlapping language for the same reasons. Ex-1003, ¶¶266-288.

Each combination further teaches the "plurality" of sandboxes in claim 15. Grounds 1-4 teach a plurality of sandboxes. *Supra* §V.A.4[10c], §V.B.3[10c] (incorporated by reference). Tarafdar performs analytics for resources in the

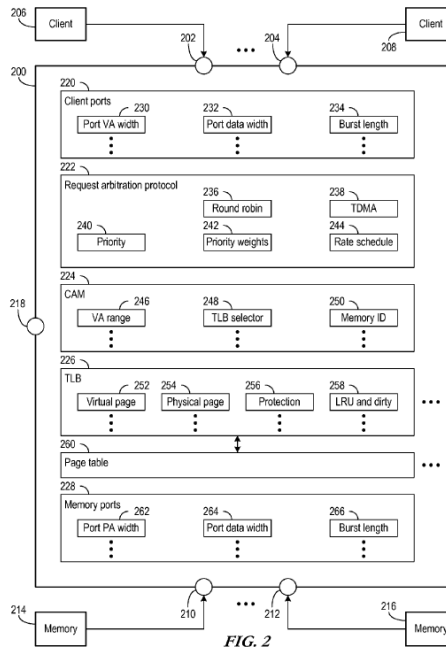
plurality of sandboxes because Tarafdar teaches partitioned FPGAs like that of Seshadri, Atsatt-113, and Biederman. Ex-1007, 000003. When the Application Region is partitioned into smaller user regions, performing the above-described analytics on the FPGA includes doing so in the plurality of user regions. Ex-1007, 000007-10; Ex-1003, ¶266-267, ¶279.

D. Ground 6

Claims 7 and 16 are obvious based on each of Grounds 1-4 in view of Kulkarni.

1. U.S. Patent No. 8,015,386 (“Kulkarni”)

Kulkarni is §§102(a)(1)-(2) art, issued/published on September 6, 2011. Ex-1003, ¶¶290-291. Kulkarni is directed to a “configurable memory manager” with port interfaces that are configurable for data-width. Ex-1008, Abs., 2:25-37, 1:12-54.



Ex-1003, ¶¶291-293.

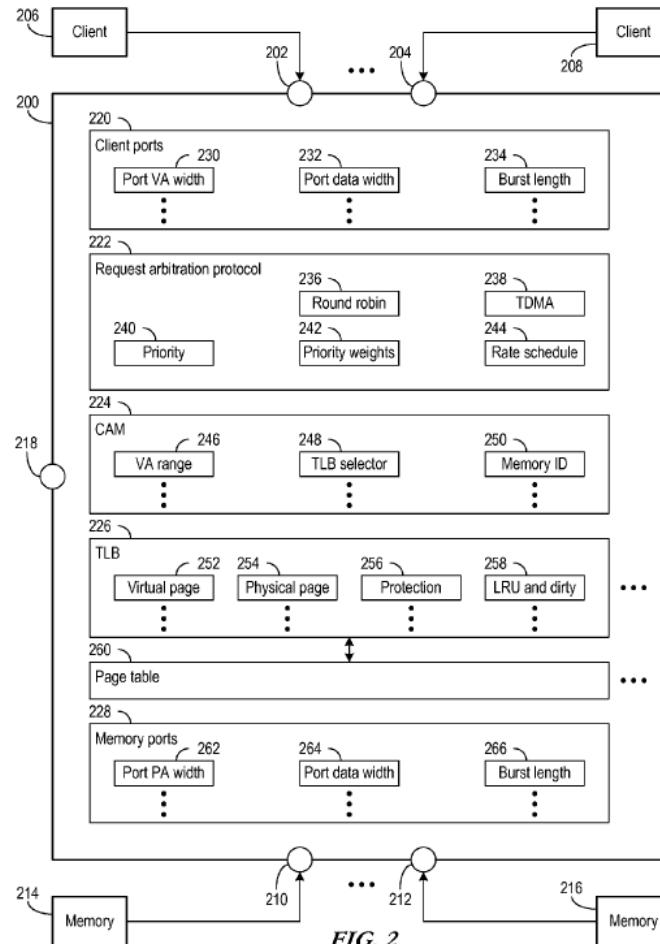
Kulkarni is analogous art in the same field of endeavor and reasonably pertinent to the '943 patent (*supra* §III), because it aims to improve the “performance of multiprocessing system[s]” implemented in a PLD with techniques applicable to interfacing communication between FPGA partitions. Ex-1008, 1:39-47; Ex-1003, ¶294.

2. Claim 7

7. The programmable device of claim 1, wherein the one or more sandboxes each comprise port interfaces that are configurable as to data width or speed.

Kulkarni teaches claim 7. Ex-1003, ¶¶295-314.

Kulkarni discloses “a configurable memory manager” with ports on portions of the programmable fabric (called “clients”) to interface between the programmable logic areas and memory in an FPGA. Ex-1008, Abs. Figs. 1-2:



The ports (and corresponding client ports) are “independently configurable to specify the...data width of the port.” Ex-1008, Abs., 2:25-37 (“data widths of ports...are configurable”); Ex-1003, ¶¶298-299.

Kulkarni explains “processing elements” in programmable logic sectors are “connected to client ports 102 through 104” and “memory access requests” are

switched to “the appropriate one of memory ports 106 through 108.” Ex-1008, 2:25-37, 1:58-2:3, Fig. 1; Ex-1003, ¶¶299-300.

Kulkarni teaches adjusting the data width for the ports on the side of the clients, because it teaches configuring “the width of read data and write data” for the corresponding ports on the configurable memory manager side. Ex-1008, 4:26-42, 6:22-27 (discussing matching settings on the memory side, Fig. 2. Therefore, each client sandbox port has a configurable data width. *See id*; Ex-1003, ¶301.

Motivation to Combine. Seshadri, Atsatt-113, and Biederman teach known techniques and uses of PLDs, including FPGAs. *Supra* §V.A.1-2, §V.B.1. A POSITA would have been motivated to further apply Kulkarni’s teachings because Kulkarni suggests using its configurable memory manager in PLDs, including FPGAs. Ex-1008, 1:11-54. Kulkarni explains this would save time and expense over “designing [custom] protocols for exchanging data.” Ex-1008, 1:48-54. Therefore, Kulkarni’s teachings were readily applicable to other PLD teachings such as Seshadri, Atsatt-113, and Biederman. Ex-1003, ¶302, ¶309.

A POSITA would have been motivated because Kulkarni provides a powerful and flexible solution for organizing communication between different parts of a PLD. Kulkarni’s configurable memory manager is particularly well-suited for FPGAs hosting a variable number of user partitions, such as in the

combinations with Seshadri, Atsatt-113, and Biederman. Ex-1008, 2:25-37, 4:26-42, 6:22-27. A POSITA would have been motivated to apply Kulkarni's teachings to adjust the data width of ports in each sandboxed partition based on the user's custom logic. *See id*; Ex-1003, ¶303, ¶310.

Moreover, Seshadri, Atsatt-113, and Kulkarni teach FPGAs with client-configurable areas. Therefore, a POSITA would have been motivated to apply Kulkarni's teachings regarding "client" areas in the FPGA because it would have used Kulkarni's configurable memory manager to improve similar devices in the same way. Ex-1008, 2:25-37, 4:26-42, 6:22-27; Ex-1003, ¶304, ¶311.

For Grounds 1-2, Seshadri provides motivation to combine with known communication modules by teaching the use of "[h]ard macros" for "communication ports." Ex-1004, 10:48-65. Macros were used to import a design into an IC. Kulkarni teaches a hard macro for communication ports that was operable with various FPGA designs. Ex-1008, 1:56-3:5, 2:58-3:12; *supra* §I.A; Ex-1003, ¶305.

For Grounds 3-4, Atsatt provides motivation to "include transceiver circuitry." Ex-1006, ¶30, ¶104 ("[T]he disclosure is not intended to be limited to the particular forms disclosed."). Kulkarni teaches an optimization for I/O circuitry, which a POSITA would have been motivated to combine. Ex-1008, 1:56-3:5, 2:58-3:12; Ex-1003, ¶311.

Kulkarni comes from Xilinx, which a POSITA would have looked to because Xilinx was widely recognized as a pioneer and market leader in FPGAs. A POSITA would have naturally looked to Xilinx patents like Kulkarni for improvements in FPGAs. Ex-1003, ¶306, ¶312.

A POSITA would have had a reasonable expectation of success because Kulkarni's teachings were meant to be added to known FPGA configurations (Ex-1008, 1:56-3:5, 2:58-3:12), and Seshadri and Atsatt-113 teach the use of such additions. Ex-1004, 10:48-65; Ex-1006, ¶30. Therefore, the combinations would not have changed the principle of operation of any reference. Ex-1003, ¶307, ¶313. Moreover, Kulkarni provides the details of how to implement its configurable memory manager on an FPGA. *E.g.*, Ex-1008, 6:28-7:41, Figs. 2-3. A POSITA would have understood Kulkarni's teachings to be additions that could be implemented with routine engineering, and would have been fully capable of making the combination. *Supra* §III; Ex-1003, ¶308, ¶314.

3. Claim 16

Claim 16 is nearly identical to claim 7. Kulkarni teaches the overlapping limitations for the same reasons. Ex-1003, ¶295-301.

Each combination further teaches the "plurality" of sandboxes in claim 16. Grounds 1-4 teach a plurality of sandboxes. *Supra* §V.A.4[10c], §V.B.3[10c] (incorporated by reference). Kulkarni teaches ports for multiple client sandboxes,

and therefore the above teachings equally to each sandboxed user partition in a plurality of partitions. Ex-1008, Fig. 2; Ex-1003, ¶296, ¶¶298-301.

E. Ground 7

Claims 8 and 17 are obvious based on each of Grounds 1-4 in view of Atsatt-923.

1. U.S. Patent Application Publication No. 2019/0041923A1 (“Atsatt-923”) (Ex-1009)

Atsatt-923 is §§102(a)(2) art filed December 27, 2017. Ex-1003, ¶¶316-317.

Atsatt-923 is directed to PLDs that use voltage regulators for “dynamic voltage frequency scaling” to manage power consumption and thermals for groups of logic sectors. Ex-1009, ¶¶32-33, ¶1, ¶18-24, Fig. 1:

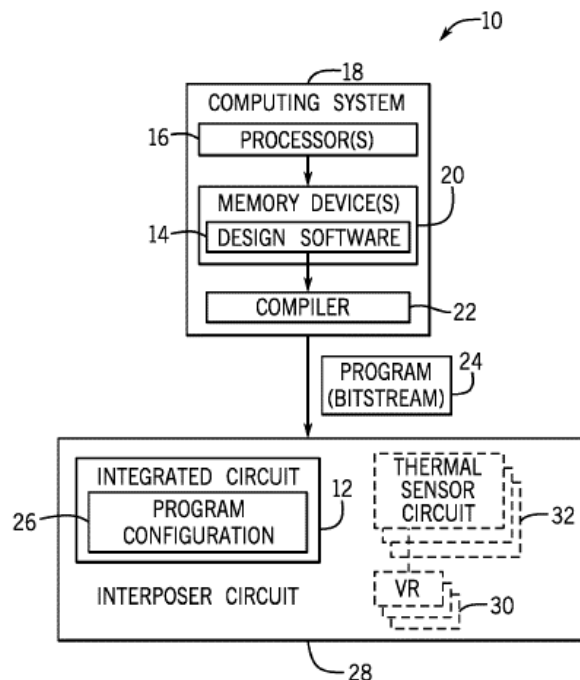


FIG. 1

Ex-1003, ¶¶317-319.

Atsatt-923 is analogous art in the same field of endeavor and reasonably pertinent to the '943 patent (*supra* §III), because it presents techniques for use with “programmable logic devices [that] are increasingly used as accelerators for machine learning, video processing, voice recognition, image recognition,” etc. to solve technical challenges arising from concentrating FPGAs in data centers. Ex-1009, ¶4; Ex-1003, ¶320.

2. Claim 8

8. The programmable device of claim 1, wherein the one or more sandboxes each comprise clock cycle utilization regulation for controlling power consumption or thermal characteristics.

Atsatt-923 teaches claim 8. Ex-1003, ¶¶321-334. For example, Atsatt-923 discloses regulating power consumption and operating temperature for groups of logic sectors in an FPGA by dynamically adjusting voltage and frequency. Ex-1009, ¶¶21-23, ¶26 (“improved thermal management”). Atsatt-923 achieves this with supporting circuits, including voltage regulators and thermal sensors. Ex-1009, ¶¶46-48, Figs. 4-9. Atsatt-923 regulates clock cycle utilization by adjusting clock cycle frequency²² and power consumption. Ex-1009, ¶62 (“use more power

²² PO alleged that “scaling down processing speed/frequency” satisfies this limitation. Ex-1021, 000058; *supra* §II.C.

(e.g., operate at higher voltage or *frequency*) [or] operate with less power.”).

Clock frequency is the number of clock cycles per second. Therefore, Atsatt-923 regulates the utilization of clock cycles by the FPGA. Additionally, Atsatt-923 regulates clock cycles utilization for a given workload because increasing clock frequency decreases clock-cycle utilization (as a percentage of total cycles). Ex-1003, ¶324.

Atsatt-923 teaches each sandbox including this clock-cycle regulation functionality because it applies frequency scaling to groups of programmable logic sectors (Ex-1009, ¶¶21-23, ¶26, ¶¶63-64), which aligns with Atsatt-113’s, Seshadri’s, and Biederman’s sandboxed partitions. Additionally, it would have been obvious to apply these teachings on a per-partition basis because Atsatt-923 aims to provide this feature for distinct sectors or portions of an FPGA (*id.*) and because clock cycle utilization would be adjusted on a per-partition basis to meet power consumption or thermal constraints each partition. Ex-1003, ¶325.

Motivation to Combine. Seshadri, Atsatt-113, and Biederman teach techniques and uses of PLDs, including FPGAs. *Supra* §V.A, §V.B. A POSITA would have been motivated to apply Atsatt-923’s teachings because Atsatt-923 suggests improving power management of “programmable logic device[s], such as a field programmable gate array (FPGA).” Ex-1009, ¶1. Thus, applying Atsatt-923’s teachings to known PLDs, including FPGAs as taught by Seshadri, Atsatt-

113, and/or Biederman, would have been a combination of prior art elements according to known methods to yield the predictable results of monitoring operational temperatures and adjusting the clock frequency for PLD/FPGA partitions, as taught by Atsatt-923. Ex-1003, ¶326, ¶330.

A POSITA would have been motivated to combine because Seshadri, Atsatt-113, and Biederman contemplate the use of PLDs in high density servers, where the need to manage temperature levels to meet Thermal Design Power (TDP) budgets was well-known. *See, e.g.*, Ex-1034, ¶¶3-4. Atsatt-923's teachings provide a solution to this problem, conveniently embedded in the PLD, for providing more efficient utilization and reduce wasted power and heat. The teachings were advantageous in terms of saving time developing external solutions for thermal management. Moreover, at the scale used in cloud computing servers, adjusting internal clock cycle utilization could also provide meaningful reductions in energy use. *See* Ex-1009, ¶5; Ex-1003, ¶327, ¶332.

For Grounds 1-2, Seshadri suggests "various modifications and changes" (Ex-1004, 29:62-30:11); a POSITA would have been motivated to apply Atsatt-923's teachings in the same spirit. Ex-1003, ¶326.

For Grounds 3-4, a POSITA would also have been motivated because Atsatt-113 and Atsatt-923 both come from Intel and share common inventors Atsatt and Weber. Ex-1006, (71)-(72); Ex-1009, (71)-(72). The two references

teach similar FPGA configurations and the teachings were meant to be used together. Ex-1006, Fig. 2; Ex-1009, Fig. 3. Biederman's additional teachings do not disturb the basic common ground of Atsatt-923 and Atsatt-113. Ex-1003, ¶331.

A POSITA would have had a reasonable expectation of success in the combinations, which would not have changed the principle of operation of any reference. Atsatt-923's teachings are simply used to adjust the voltage and speed of a PLD/FPGA in the manner taught by Atsatt-923. As Atsatt-923 explains, implementing its disclosures would "be a routine undertaking...for those of ordinary skill having the benefit of this disclosure." Ex-1009, ¶16. And the management of power, voltage, and frequency was well-known in data centers and for FPGAs. Therefore, the combinations simply use each reference for its intended purpose, without requiring extensive modifications. Atsatt-923's voltage/frequency-scaling teachings were additions, to be added to existing FPGA designs, that could be implemented with routine engineering, and a POSITA would have been fully capable of combining the teachings. *Supra* §III; Ex-1003, ¶¶328-329, ¶¶333-334.

3. Claim 17

Claim 17 is nearly identical to claim 8. Atsatt-923 teaches the overlapping language for the same reasons. Ex-1003, ¶¶321-329.

Each combination further teaches the “plurality” of sandboxes in claim 15. Grounds 1-4 teach a plurality of sandboxes. *Supra* §V.A.4[10c], §V.B.3[10c] (incorporated by reference). Atsatt-923 teaches dynamic frequency/voltage regulation for each of one or more sandboxes in an FPGA; therefore it also teaches the same for each of a plurality of sandboxed user partitions. Ex-1009, Fig. 2; Ex-1003, ¶¶321-325.

VI. STANDING

Petitioner certifies that Petitioner is not barred or estopped from requesting this IPR and the '943 patent is eligible for it.

VII. DISCRETIONARY ANALYSIS

Discretionary denial is unwarranted. This Petition’s merits are compelling, which “alone demonstrates that the PTAB should not discretionarily deny institution under *Fintiv*.” *See* June 21, 2022 Director’s Memo, 3-5. The Grounds present compelling obviousness theories that, “if unrebutted in trial, would plainly lead to a conclusion that one or more claims are unpatentable by a preponderance of the evidence.” *Id.*, 4.

Fintiv. The factors favor institution. The district court case is in its early stages: claim construction has not been briefed. The trial date is currently May 18, 2026. The Court has not ruled on Petitioner’s pending motions to transfer and dismiss; hearings have not been scheduled.

Advanced Bionics. Denial under §325(d) is unwarranted because Seshadri, Biederman, Atsatt-113, Tarafdar, Kulkarni, and Atsatt-923 were not cited or considered during prosecution. This is the only IPR filed against the challenged claims to date. *General Plastic* factors do not weigh against institution.

VIII. MANDATORY NOTICES AND FEES

1. **Real Party In Interest**: The real parties-in-interest here are Advanced Micro Devices, Inc. and Pensando Systems, Inc. No other parties directed, controlled, or funded this Inter Partes Review proceeding (IPR).

2. **Related Matters**:

- *Concurrent Ventures, LLC et al. v. Advanced Micro Devices, Inc. et al.*, Case No. 1-24-cv-00335 (W.D. Tex.)

3. **Lead and Backup Counsel**: Pursuant to 37 C.F.R. §42.8(b)(3) and 42.10(a), Petitioner designates Harper Batts (Reg. No. 56,160) as lead counsel and Chris Ponder (Reg. No. 77,167) and Jeffrey Liang (Reg. No. 69,043) as back-up counsel, each of Sheppard, Mullin, Richter & Hampton LLP.

4. **Service Information**: Petitioner consents to service by electronic mail addressed to:

HBatts@sheppardmullin.com;
CPonder@sheppardmullin.com;
JLiang@sheppardmullin.com; and
LegalTM-AMD-Concurrent@sheppardmullin.com

5. Petitioner's counsel may also be served by mail or hand delivery at Sheppard, Mullin, Richter & Hampton LLP, 1540 El Camino Real Suite 120, Menlo Park, CA 94025. Petitioner's counsel may be reached by telephone at (650) 815-2673 and by facsimile at (650) 815-4668.

6. Fees: The Office is authorized to charge fees for this Petition to Deposit Account 50-4561, Ref. 71EY-388593.

Respectfully submitted,

SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/Harper Batts/

Harper Batts (Reg. No. 56,160)
Attorneys for Petitioner

CLAIM APPENDIX

[1pre]. A programmable device for data flow processing in a user-configurable server with swappable pods or cards, comprising:

[1a] a programmable logic device (PLD) having a first region and a second region;

[1b] the first region comprising a hardware-based or firmware-based router with a port enabling communication between the router and the second region; and

[1c] the second region comprising one or more sandboxes with user-definable programmable electronic circuits of the PLD,

[1d] wherein the port comprises: a bridge being lockable to prevent user access, and unlockable to enable user access, to each of the sandboxes and the user-definable programmable electronic circuits therein.

2. The programmable device of claim 1, further comprising:

the router communicative to translate between a first protocol and a second protocol, for messaging with system resources.

3. The programmable device of claim 1, wherein each of the sandboxes comprises user-programmable one or more of: analog circuitry, digital circuitry, memory, processors or controllers.

4. The programmable device of claim 1, wherein the port comprises:

a bridge being lockable to prevent reconfiguration of the bridge and prevent reconfiguration of the router while enabling reconfiguration of the one or more sandboxes.

5. The programmable device of claim 1, wherein the port comprises:

a bridge having a security layer to authenticate a user or verify a permission, to reconfigure one or more of the one or more sandboxes.

6. The programmable device of claim 1, further comprising:

the router configured to perform analytics on resources of the programmable device, on a per module basis, including determining communication bandwidth and resource utilization.

7. The programmable device of claim 1, wherein the one or more sandboxes each comprise port interfaces that are configurable as to data width or speed.

8. The programmable device of claim 1, wherein the one or more sandboxes each comprise clock cycle utilization regulation for controlling power consumption or thermal characteristics.

9. The programmable device of claim 1, wherein the one or more sandboxes comprises a plurality of sandboxes that are logically or physically combinable to form a larger sandbox.

10[pre]. A programmable device for data flow processing in a user-configurable server, comprising:

[10a] a programmable logic device (PLD) having a first region and a second region;

[10b] the first region comprising a hardware-based or firmware-based router having a bridge with a port enabling communication between the router and the second region;

[10c] the second region comprising a plurality of sandboxes with user-definable programmable electronic circuits of the PLD; and

[10d] an internal communication network in the PLD, coupling the port of the bridge of the router and the one or more sandboxes for the data flow processing,

[10e] wherein the bridge is lockable to prevent user access, and unlockable to enable user access, to the plurality of sandboxes and user-definable programmable electronic circuits therein.

11. The programmable device of claim 10, further comprising:

the router to translate between a first protocol used external to the second region and a second protocol used in the internal communication network in the PLD.

12. The programmable device of claim 10, wherein each of the plurality of sandboxes comprises user-programmable one or more of: analog circuitry, digital circuitry, memory, processors or controllers.

13. The programmable device of claim 10, wherein the bridge is lockable to enable reconfiguration of the plurality of sandboxes, prevent reconfiguration of the bridge and prevent reconfiguration of the router.

14. The programmable device of claim 10, wherein the bridge has a security layer to authenticate a user or verify a permission, to reconfigure one or more of the plurality of sandboxes.

15. The programmable device of claim 10, further comprising:
the router configured to perform analytics on resources of the programmable device, on a per module basis, including determining communication bandwidth and resource utilization in the plurality of sandboxes.

16. The programmable device of claim 10, wherein the each of the plurality of sandboxes comprises one or more port interfaces configurable as to data width or speed.

17. The programmable device of claim 10, wherein each of the plurality of more sandboxes comprises clock cycle utilization regulation for controlling power consumption or thermal characteristics.

18. The programmable device of claim 10, wherein the plurality of sandboxes is logically or physically combinable to form a single, larger sandbox.

Side-by-Side List of Challenged Claims (differences in bolded green):

[1pre]. A programmable device for data flow processing in a user-configurable server with swappable pods or cards , comprising:	10[pre]. A programmable device for data flow processing in a user-configurable server, comprising:
[1a] a programmable logic device (PLD) having a first region and a second region;	[10a] a programmable logic device (PLD) having a first region and a second region;
[1b] the first region comprising a hardware-based or firmware-based router with a port enabling communication between the router and the second region; and	[10b] the first region comprising a hardware-based or firmware-based router having a bridge with a port enabling communication between the router and the second region;
[1c] the second region comprising one or more sandboxes with user-definable programmable electronic circuits of the PLD,	[10c] the second region comprising a plurality of sandboxes with user-definable programmable electronic circuits of the PLD; and
	[10d] an internal communication network in the PLD, coupling the port of the bridge of the router and the one or more sandboxes for the data flow processing,

[1d] wherein the port comprises: a bridge being lockable to prevent user access, and unlockable to enable user access, to each of the sandboxes and the user-definable programmable electronic circuits therein.	[10e] wherein the bridge is lockable to prevent user access, and unlockable to enable user access, to the plurality of sandboxes and user-definable programmable electronic circuits therein.
---	---

2. The programmable device of claim 1 , further comprising: the router communicative to translate between a first protocol and a second protocol, for messaging with system resources .	11. The programmable device of claim 10 , further comprising: the router to translate between a first protocol used external to the second region and a second protocol used in the internal communication network in the PLD .
3. The programmable device of claim 1 , wherein each of the sandboxes comprises user-programmable one or more of: analog circuitry, digital circuitry, memory, processors or controllers.	12. The programmable device of claim 10 , wherein each of the plurality of sandboxes comprises user-programmable one or more of: analog circuitry, digital circuitry, memory, processors or controllers.
4. The programmable device of claim 1 , wherein the port comprises: a bridge being lockable to prevent reconfiguration of the bridge and prevent reconfiguration of the router while enabling reconfiguration of the one or more sandboxes.	13. The programmable device of claim 10 , wherein the bridge is lockable to enable reconfiguration of the plurality of sandboxes, prevent reconfiguration of the bridge and prevent reconfiguration of the router.

5. The programmable device of claim 1 , wherein the port comprises: a bridge having a security layer to authenticate a user or verify a permission, to reconfigure one or more of the one or more sandboxes.	14. The programmable device of claim 10 , wherein the bridge has a security layer to authenticate a user or verify a permission, to reconfigure one or more of the plurality of sandboxes.
6. The programmable device of claim 1 , further comprising: the router configured to perform analytics on resources of the programmable device, on a per module basis, including determining communication bandwidth and resource utilization.	15. The programmable device of claim 10 , further comprising: the router configured to perform analytics on resources of the programmable device, on a per module basis, including determining communication bandwidth and resource utilization in the plurality of sandboxes .
7. The programmable device of claim 1 , wherein the one or more sandboxes each comprise port interfaces that are configurable as to data width or speed.	The programmable device of claim 10 , wherein the each of the plurality of sandboxes comprises one or more port interfaces configurable as to data width or speed.
8. The programmable device of claim 1 , wherein the one or more sandboxes each comprise clock cycle utilization regulation for controlling power consumption or thermal characteristics.	The programmable device of claim 10 , wherein the each of the plurality of sandboxes comprises clock cycle utilization regulation for controlling power consumption or thermal characteristics.

9. The programmable device of claim 1, wherein the one or more sandboxes comprises a plurality of sandboxes that are logically or physically combinable to form a larger sandbox.	18. The programmable device of claim 10, wherein the plurality of sandboxes is logically or physically combinable to form a single , larger sandbox.
---	---

CERTIFICATE OF SERVICE

The undersigned hereby certifies that on December 19, 2024, true and correct copies of the foregoing Petition for *Inter Partes* Review of U.S. Patent No. 10,985,943 and Exhibits 1001-1034 were served in entirety on the Patent Owner at the following addresses of record as listed on Patent Center, via Priority Mail Express®:

Michael Gencarella
12875 Lucero Lane
Los Altos Hills, CA 94022
Official Correspondence Address

Concurrent Ventures LLC
c/o Beeson, Jesse D.
240 Woodscape Court
Johns Creek, GA, 30022-3244
Patent Owner Correspondence Address

XtreamEdge, Inc.
c/o The Corporation Trust Company
Corporation Trust Center
1209 Orange Street
Wilmington, DE 19801
Patent Owner Correspondence Address

Dated: December 19, 2024

/Harper Batts/

Harper S. Batts (Reg. No. 56,160)
Attorney for Petitioner

1540 El Camino Real Suite 120
Menlo Park, CA 94025
Tel: (650) 815-2600

The undersigned certifies that true and correct copies of the Petition for *Inter Partes* Review of U.S. Patent No. 10,985,943 and Exhibits 1001-1034 were also sent via electronic mail to the attorneys of record for Plaintiff in the concurrent litigation matter(s):

Michael Matulewicz-Crowley
mmatulewicz-
crowley@reichmanjorgensen.com
Christine E. Lehman
clehman@reichmanjorgensen.com;
myohannes@reichmanjorgensen.com

Ariane S. Mann
amann@reichmanjorgensen.com
Amy L. Ruhland
aruhland@reichmanjorgensen.com
Laura Carwile
lcarwile@reichmanjorgensen.com

SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/Harper Batts/

Harper Batts (Reg. No. 56,160)
Attorneys for Petitioner

Date: December 19, 2024

1540 El Camino Real Suite 120
Menlo Park, CA 94025
Tel: (650) 815-2673

**CERTIFICATE OF COMPLIANCE WITH
TYPE-VOLUME LIMITATION, TYPEFACE REQUIREMENTS,
AND TYPE STYLE REQUIREMENTS**

1. This Petition complies with the type-volume limitation of 14,000 words, comprising 13,620 words, as counted using the Microsoft Word software that was used to prepare this paper, excluding the parts exempted by 37 C.F.R. § 42.24(a).

2. This Petition complies with the general format requirements of 37 C.F.R. § 42.6(a) and has been prepared using Microsoft® Word in 14-point Times New Roman.

Respectfully submitted,

SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/Harper Batts/
Harper Batts (Reg. No. 56,160)
Attorney for Petitioner

Date: December 19, 2024

1540 El Camino Real Suite 120
Menlo Park, CA 94025
Tel: (650) 815-2673