

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

INTEL CORPORATION,
Petitioner,

v.

BiTMICRO, LLC,
Patent Owner.

Case IPR2023-00787
Patent 10,120,694

PATENT OWNER'S PRELIMINARY RESPONSE

TABLE OF CONTENTS

I.	INTRODUCTION	1
A.	The Disclosed Invention Leverages a POR Sequencer for the Boot Process	2
B.	The Challenged Claims and Petitioner’s Challenges	6
II.	THE BOARD NEED NOT ENGAGE IN CLAIM CONSTRUCTION TO DENY THE PETITION	10
III.	PETITIONER’S CHALLENGES FAIL ON THE MERITS	11
A.	Ground 1 based on Post with Kao and Wilson Fails.....	11
i.	Petitioner’s Proposed Combination of Post with Kao is Unsupported by Either Reference, and Fails to Disclose Every Claimed Feature.....	11
ii.	Petitioner’s Proposed Combination of Post, Kao and Wilson Fails to Disclose or Suggest a POR Sequencer that Verifies Integrity of Loaded Data and Code in the RAM, as Claimed.....	15
iii.	Petitioner’s Proposed Combination Fails to Disclose or Suggest a POR Sequencer Descriptor that Includes DMA Controller Descriptors, as Claimed	18
iv.	Ground 1’s Challenge Fails to Establish the Unpatentability of Claim 1	21
B.	Ground 2 based on Post, Kao, Wilson, and So Also Fails.....	22
IV.	CONCLUSION.....	23

Pursuant to 37 C.F.R. § 42.107, Patent Owner BiTMICRO, LLC (“BiTMICRO” or “Patent Owner”) files this preliminary response to the Petition, setting forth reasons why the Petition for *inter partes* review (“IPR”) of claims 1, 5, 6, 9, and 12 of U.S. Patent No. 10,120,694 (the “’694 Patent”), as requested by Intel Corporation (“Petitioner”), should be denied.¹

I. INTRODUCTION

The ’694 Patent, entitled “Embedded System Boot from a Storage Device,” relates to an apparatus and mechanism for booting an embedded system from a storage device, “prior to activating a CPU.” ’694 Pat., Abstr. To achieve benefits in terms of cost and complexity, the ’694 Patent describes an exemplary apparatus that includes a power-on reset (POR) sequencer that “controls the overall boot process of the embedded system” from nonvolatile memory prior to activating a CPU. ’694 Pat., 2:35-36. The POR sequencer “uses descriptors ... which are used to update the configuration registers of the system and to enable CPU-independent data transfers with the use of DMA controllers.” ’694 Pat., Abstr. The POR sequencer’s loading of DMA controller descriptors enables the initialization of associated DMA controllers, which can then take control of the CPU bus to

¹ By submitting this Preliminary Response, no waiver of any argument is intended by Patent Owner. Patent Owner will have a right to file a complete response to the Petition, should the Board institute *inter partes* review. 37 C.F.R. § 42.120(a).

perform memory transactions independently of the CPU and therefore conserve processing resources for other activities.

A. The Disclosed Invention Leverages a POR Sequencer for the Boot Process

As illustrated in FIG. 2 of the '694 Patent, an exemplary embedded system includes one or more central processing units (CPUs), a reset controller, a storage device controller, one or more direct memory access (DMA) controllers, a random access memory (RAM), a nonvolatile memory controller, a debug interface, and a POR sequencer. *See, e.g., '694 Pat., Fig. 2.*

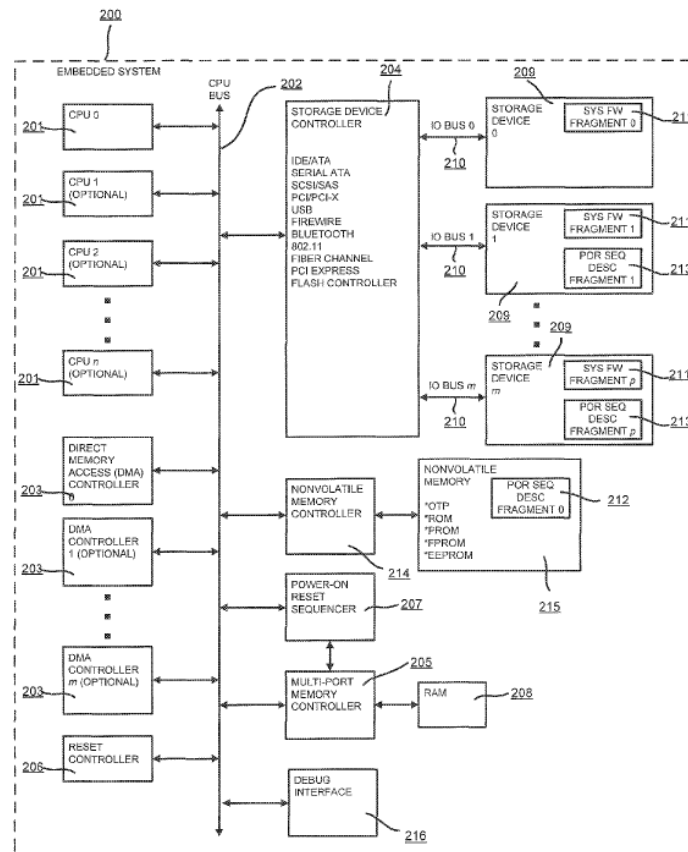


FIG. 2

Although conventional embedded systems use a CPU to initiate and/or control the boot process, the '694 Patent uses the POR sequencer to detect and load a POR sequencer descriptor prior to initializing the CPU. As illustrated in Figure 4 of the '694 Patent, reproduced below, an exemplary POR sequencer descriptor includes register information for a controller core, control information and DMA controller descriptors to activate the DMA controllers, and a POR sequencer descriptor checksum to not only ensure that the entirety of the POR sequencer descriptor is loaded into RAM, but to ensure the integrity of the loaded data and code. *See, e.g.*, '694 Pat., 6:67-7:5 (“POR sequencer 207 then checks the integrity of POR sequencer descriptor 212 [loaded into RAM 208] by comparing the computed checksum to the POR sequencer descriptor checksum 507.”).

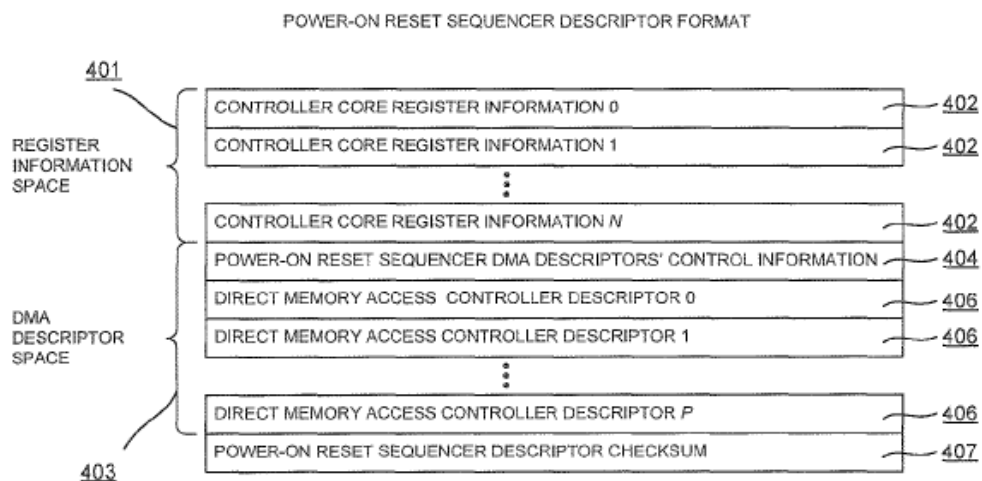
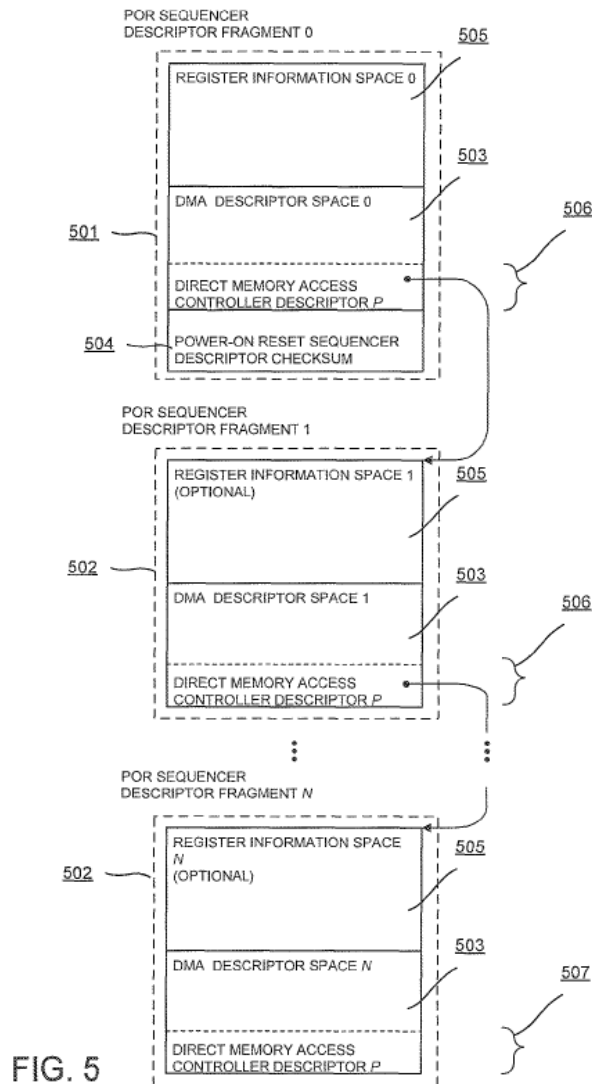


FIG. 4

In instances where the POR sequencer descriptor cannot be stored in a single block or location of the nonvolatile memory, the '694 Patent also discloses the exemplary use of POR sequencer descriptor fragments, where “the descriptor is divided into fragments and distributed across the nonvolatile memory and storage devices attached to the system.” '694 Pat., 6:32-35. In this case, the '694 Patent discloses the option to store register information space fragments, DMA descriptor space fragments, and DMA controller descriptor fragments in each POR sequencer descriptor fragment, with the POR sequencer descriptor checksum stored in the first POR sequencer descriptor fragment. In order to point to next POR sequencer descriptor fragment, the last DMA controller descriptor in a POR sequencer descriptor fragment also “points to the next POR sequencer descriptor fragment.” '694 Pat., 6:42-44. This arrangement of POR sequencer descriptor fragments is illustrated in FIG. 5, reproduced below.



Consistent with this disclosure, the challenged claims of the '694 Patent include a POR sequencer to detect a descriptor in nonvolatile memory, and to copy the descriptor from nonvolatile memory to RAM. As expressly claimed, the POR sequencer is "configured to detect, in the nonvolatile memory, a descriptor used in a boot process and configured to copy the descriptor from the nonvolatile memory to the RAM." '694 Pat., 9:20-23. This descriptor comprises a POR sequencer

descriptor, and it includes “register information and DMA controller descriptors.”

'694 Pat., 9:31-32. Additionally, the POR sequencer “verifies an integrity of loaded data and code in the RAM” to ensure the boot process can be performed successfully. '694 Pat., 9:24-26.

B. The Challenged Claims and Petitioner's Challenges

Consistent with these disclosures, the challenged claims are directed generally to an apparatus configured to perform a boot process of an embedded system using a power-on reset (POR) sequencer. Claim 1 is the focus of Petitioner's challenges, and is reproduced below:

1. An apparatus, comprising:
an embedded system comprising one or more processors, a reset controller, a storage device controller, one or more direct memory access (DMA) controllers, a random access memory (RAM) and a memory controller, a nonvolatile memory and a nonvolatile memory controller, a debug interface, and
a power-on reset (POR) sequencer configured to detect, in the nonvolatile memory, a descriptor used in a boot process and configured to copy the descriptor from the nonvolatile memory to the RAM,
wherein the POR sequencer facilitates a boot process and verifies an integrity of loaded data and code in the RAM,
wherein the descriptor comprises a POR sequencer descriptor, and
wherein the POR sequencer uses the POR sequencer descriptor

which is a preassembled descriptor that is stored in the nonvolatile memory,
wherein the POR sequencer descriptor includes register information and DMA controller descriptors, and
wherein the POR sequencer reads and uses the register information to update configuration registers of the embedded system.

’694 Pat., 9:13-35. Petitioner asserts two obviousness-based grounds against claim 1 and the additional challenged claims, claims 5, 6, 9, and 12. Of these challenged claims, claims 6 and 12 are independent, and claims 5 and 9 respectively depend from claims 1 and 6. For the purpose of Petitioner’s challenges, claims 6 and 12 are substantially similar to claim 1 as discussed in Section III below.

In the Petition, Petitioner contends that all challenged claims are obvious over a combination of the Post² reference, the Kao³ reference, and the Wilson⁴ reference, and also a combination of Post, Kao, Wilson, and the So⁵ reference. *See* Pet., 5. However, Petitioner’s challenges fail for numerous reasons.

First, Petitioner’s combination of Post and Kao is inconsistent with the teachings of those references. In order to recreate the ’694 Patent’s embedded system having at least one processor and at least one DMA controller as claimed,

² U.S. Patent No. 8,799,555 to Post *et al.* (“Post”) (Ex. 1005).

³ U.S. Patent No. 7,490,177 to Kao (“Kao”) (Ex. 1004).

⁴ U.S. Patent No. 7,861,073 to Wilson (“Wilson”) (Ex. 1013).

⁵ U.S. Patent No. 7,526,679 to So (“So”) (Ex. 10121).

Petitioner proposes to incorporate Kao's DMA engine into Post's embedded system for the purpose of reducing the processing burden on Post's processor. But in challenging the POR sequencer of claim 1, Petitioner relies on Post's processor to perform the functions of detecting and loading boot code, without acknowledging that the entire purpose for Petitioner's combination of Post and Kao was to use Kao's DMA engine for those functions. Thus, in Petitioner's combination of Post and Kao, Post's processor does not perform those functions and cannot serve as the claimed "POR sequencer" in either of Petitioner's challenges. Petitioner's argument to the contrary is flatly inconsistent with Kao's teaching regarding its use of the DMA engine upon start-up, and Petitioner's basis for modifying Post with Kao.

Second, Petitioner incorrectly relies on Post's use of pointers to allegedly disclose the claimed requirement of the POR sequencer "verif[y]ing an integrity of loaded data and code in the RAM." While Post discloses the use of pointers to ensure that every page of boot code is transferred, Post does not disclose any verification of boot code integrity once that boot code is loaded "in the RAM" as required by claim 1. Petitioner also points to Post's disclosure of error correction on individual pages, but Post is clear that this error correction is to ensure that an identified page still in non-volatile memory contains boot data, and does not verify the integrity of code loaded in RAM.

Third, Petitioner correctly acknowledges that Post fails to disclose a DMA controller, but then incorrectly asserts that Post discloses the claimed “DMA controller descriptors” as part of the claimed “POR sequencer descriptor.” Not only is this assertion wrong, but Petitioner makes the additional error of correlating the claimed “DMA controller descriptor” to Post’s pointer. While the ’694 Patent discloses that a DMA controller descriptor fragment can *include* a pointer to point to a next POR sequencer descriptor fragment, the ’694 Patent also discloses that the DMA controller descriptor is written by the POR sequencer to a “configuration space of an available DMA controller” to activate that DMA controller. ’694 Pat., 7:50-59. Petitioner overlooks this disclosure in its flawed search for the claimed “DMA controller descriptor” in Post.

Finally on this shortcoming, Petitioner also overlooks that Kao does not disclose a DMA controller descriptor either, and does not disclose copying a DMA controller descriptor to RAM in a boot-up process. Rather, Kao pre-stores information in a NAND flash memory before booting to avoid the need to load code for activating a DMA controller as part of the boot-up process. Kao’s entire purpose is to obviate the need to load DMA controller code to initialize a DMA controller upon start-up. For all these reasons, Kao fails to disclose any DMA controller descriptor to be incorporated into Post. Whether considered alone or in combination, Petitioner’s proposed combination of Post and Kao fails to disclose

any DMA controller descriptors as claimed, and all claimed use of DMA controller descriptors as part of a POR sequence descriptor “used in a boot process” per claim 1.

For these reasons, as more fully explained below, both Grounds presented in the Petition are deficient. The Petition should be denied without institution.

II. THE BOARD NEED NOT ENGAGE IN CLAIM CONSTRUCTION TO DENY THE PETITION

In the Petition, Petitioner contends that no terms require construction to “resolve the patentability disputes in this proceeding,” and purports to be applying the plain and ordinary meaning. *See* Pet., 17. However, Petitioner’s proposed characterization of the claimed “DMA controller descriptors” as satisfied by Post’s pointer is incorrect, and does not comport with the plain and ordinary meaning of this claim term. Further, this characterization fails to consider the ’694 Patent’s specification disclosure that the “DMA controller descriptors” allow for activation of associated DMA controllers. The proper interpretation of the claimed “DMA controller descriptors,” and why Petitioner is wrong to interpret this claimed term as a pointer, is addressed in detail in the context of Patent Owner’s substantive arguments on this term in Section III.A.iii below.

Without waiving its right to submit additional proposed constructions in a Patent Owner Response, should one be necessary, Patent Owner submits that the Board need not engage in any pre-institution claim construction to deny the

Petition. For the reasons presented in Section III below, Petitioner's Grounds each include fundamental flaws and fail to establish the obviousness of any challenged claim.

III. PETITIONER'S CHALLENGES FAIL ON THE MERITS

A. Ground 1 based on Post with Kao and Wilson Fails

i. Petitioner's Proposed Combination of Post with Kao is Unsupported by Either Reference, and Fails to Disclose Every Claimed Feature

As noted above, claim 1 recites, in relevant part, "an embedded system comprising one or more processors, [and] one or more direct memory access (DMA) controllers," and also "a power-on reset (POR) sequencer configured to detect, in the nonvolatile memory, a descriptor used in a boot process and configured to copy the descriptor from the nonvolatile memory to the RAM."

Petitioner concedes that Post's electronic device 100 fails to disclose any Direct Memory Access ("DMA") controller, and Petitioner relies on Kao to disclose an embedded system 100 having a DMA engine 120. Pet., 17-22. In particular, in the Petition's overview discussion of Kao, Petitioner notes that "Kao described using the DMA engine instead of the host processor to copy boot data from a nonvolatile NAND flash memory to a boot code buffer." Pet., 23 (citing Kao, 8:26-29, 8:39-45). The alleged advantage of using the DMA engine instead of a host processor to "load boot codes" was to reduce complexity and cost of the host processor and the embedded system. Pet., 23 (citing Kao, 3:7-9) (cleaned up).

The Petition’s Ground 1 challenge specifically relies on Kao’s DMA engine 120 to disclose the claimed “one or more direct memory access (DMA) controllers.” Pet., 33-38. Indeed, Petitioner spends four pages discussing reasons why a POSITA would have viewed the substitution of Kao’s DMA controller and DMA techniques for Post’s use of the “core processor to transfer boot data from nonvolatile memory to RAM” as obvious. Pet., 34-38. As a result, as Petitioner explains it, “in the resulting combination, Post’s control circuitry 112 would be modified to include a DMA controller (e.g., Kao’s DMA engine 120) that performed Post’s described functions of, at least, transferring boot data pages *subsequent to the first identified page* from nonvolatile memory 120 to memory 114.” Pet., 34 (emphasis added). However, this proposed use of Kao’s DMA engine 120 is irreconcilably inconsistent with Kao’s teachings.

In addition to requiring one or more DMA controllers in the embedded system of claim 1, claim 1 also requires, in relevant part, “a power-on reset (POR) sequencer configured to detect, in the nonvolatile memory, a descriptor used in a boot process and configured to copy the descriptor from the nonvolatile memory to the RAM.” Petitioner argues that “Post’s control circuitry 112 ... facilitated a boot process for electronic device 100 after power-on and was a ‘POR sequencer.’” Pet., 41 (citing Ex. 1003, ¶¶103-106). But this argument ignores Petitioner’s use of Kao’s DMA engine 120 in lieu of Post’s control circuitry 112 for the purpose of

transferring boot data pages from nonvolatile memory 120 to memory 114 in the relied-upon combination of references. Kao discloses that the DMA engine 120 is located “on the volatile memory chip to load boot and controller codes,” and does not need to be initialized by a host processor because its controller code is burned into NAND flash memory before booting. Kao, 2:5-15 (describing Kao’s invention as improving on the need for a host to initialize a DMA controller), 2:66-3:9, 4:49-60 (disclosing storing code in NAND flash memory before booting), 4:61-5:4 (disclosing use of the DMA engine without host initialization).

Petitioner argues that its proposed combination is only using Kao’s DMA engine 120 to transfer boot data pages *subsequent to the first identified page*, and is still using Post’s control circuitry 112 to detect and transfer the first identified page. Pet., 34. Petitioner’s theory requires Post’s control circuitry 112 to first begin the detection and transfer of a first page of boot code, and then hand over the subsequent transfer duties to the DMA engine 120. But this theory deviates from any teachings of the asserted references, and is not supported by either Post or Kao. Specifically, Kao teaches a boot sequence where the DMA engine 120 performs “automatic transfer of the boot code 132 from the NAND flash memory 130 to the boot code buffer 212 without external control or input, e.g., by the host 102 or other external controller circuitry.” Kao, 4:66-5:3. Petitioner’s staggered process negates the very advantages of using the DMA engine 120 that Kao

discloses, and undercuts Petitioner's very reason for adding DMA engine 120 to Post in the first place, namely the reduced cost and complexity of the control circuitry 112.

Indeed, contrary to Petitioner's contention, Kao discloses the "DMA engine 120 ... automatically transfer boot codes 132 and controller codes 134 ... for use in initializing and operating the host 102 and embedded microcontroller 118." Kao, 8:61-67. Thus, Kao's teaching prioritizes initialization of the DMA engine 120 over a host 102/microcontroller 118, and only initializes the host 102/microcontroller 118 after the DMA engine 120 has completed an initial transfer of boot codes 132 and controller codes 134. Petitioner's proposed inclusion of Kao's DMA engine 120 into Post for the benefit of reducing the burden on control circuitry 112, followed by Petitioner's rejection of this very benefit in its attempt to recreate the challenged claim, is unsupported by the evidence of record.

Petitioner's adoption of Kao's DMA engine 120 for the purported benefits of that DMA engine 120 as discussed in the Petition (*see* Pet., 34-38) confirms that Post's control circuitry 112 would not "detect, in the nonvolatile memory, a descriptor used in a boot process and configured to copy the descriptor from the nonvolatile memory to the RAM," as claimed. Indeed, Petitioner's cobbled-together theory of obviousness fails to acknowledge or address why a POSITA

would have been motivated to include a POR sequencer in an embedded system that already includes a processor as part of Post’s control circuitry 112,⁶ and a DMA controller, such as Kao’s DMA engine 120.

For all these reasons, Petitioner’s combination is not supported by any evidence of record, and the resulting combination fails to disclose a POR sequencer performing the claimed functions as recited in claim 1.

ii. Petitioner’s Proposed Combination of Post, Kao and Wilson Fails to Disclose or Suggest a POR Sequencer that Verifies Integrity of Loaded Data and Code in the RAM, as Claimed

Claim 1 also recites, in relevant part, “wherein the POR sequencer facilitates a boot process and verifies an integrity of loaded data and code *in the RAM*.” (emphasis added). The Petition fails to establish that these features are present or suggested by Petitioner’s proposed combination of Post, Kao, and Wilson.

Petitioner alleges that Post’s “Control circuitry 112 verified an integrity of the boot data that was loaded into memory 114 ... by using one or more of next page pointers 302, prior page pointers 304, and boot data page indicators 306.” Pet., 55 (citing Post, 6:54-55, 7:21-23, 7:58-64, 1:55-2:3, 6:22-24, 8:52-66, 9:33-

⁶ In claim 1, the “one or more processors” are recited separately from the claimed “power-on reset (POR) sequencer” using different terms, and are therefore presumed to have different meanings. *SimpleAir, Inc. v. Sony Ericsson Mobile Commc’ns AB*, 820 F.3d 419, 431(Fed. Cir. 2016); *Bd. of Regents of the Univ. of Tex. Sys. v. BENQ Am. Corp.*, 533 F.3d 1362, 1371 (Fed. Cir. 2008) (“Different claim terms are presumed to have different meanings.” (citation omitted)).

46, 10:52-54). However, this argument includes two fatal flaws. First, the use of pointers in Post does not “verif[y] an integrity of *loaded data and code in the RAM*,” as required by claim 1. (emphasis added). Second, the use of pointers in Post is designed to ensure that every page of boot data is accounted for in the nonvolatile memory, but does not verify that the data on the pages is accurate.

The Petition concedes that Post’s prior page pointers and next page pointers “may allow an electronic device to verify and account for all of the boot data *stored in the NVM*.” Pet., 55 (quoting Post, 7:21-23) (emphasis added). By looking forward and back using the page pointers, Post can account for all of the pages associated with boot data that are stored in the non-volatile memory. Post teaches that the pointers “verify that pages expected to have boot data actually contain boot data (and vice versa).” Post, 6:22-24. However, the claimed requirement is to verify the integrity of the data and code *loaded in the RAM*, not boot data still stored in non-volatile memory. Post’s pointers do not allow for any verification of the integrity of the data and code after it has been loaded from nonvolatile memory to RAM.

Further, the existence of pointers in Post does not provide any verification of data integrity in Post’s pages. Petitioner appears to recognize this defect of Post’s pointers, and also discusses Post’s use of error correcting codes “on the contents of page 2.” Pet., 55-56; Post, 6:57-59. While Petitioner contends that this use of

error correcting codes on the contents of page 2 occurs on page 2 once loaded in memory 114, Post does not support this interpretation. Pet., 55. Rather, according to Post, the use of error correcting codes is on data that has not been loaded out of nonvolatile memory. Post discloses that the error correction occurs on page 2, which is in a block “in the boot data storage area of NVM 120,” or nonvolatile memory, not RAM as required by claim 1. Post, 6:21-24, 6:54-59.

Additionally, Post explains that this use of error correcting codes is to “confirm that page 2 provides boot data,” not to check the page for errors. Post, 6:58. Post states that the determination of whether a page contains boot data “can be performed by employing an error correcting code on the initial page data *to check for a valid codeword*, or by checking a program status of the initial page to determine whether the initial page is programmed or unprogrammed.” Post, 8:34-40 (emphasis added). This process is performed on a page-by-page basis. As already noted, Post discloses that the error correction occurs on page 2, which is in a block “in the boot data storage area of NVM 120,” or nonvolatile memory. Post, 6:21-24, 6:54-59. If this process results in “uncorrectable errors” on the page, Post can still find another copy of the boot data “elsewhere.” Post, 6:60-64.

For all these reasons, Petitioner's combination of Post, Kao, and Wilson fails to disclose or suggest at least “wherein the POR sequencer facilitates a boot process and verifies an integrity of loaded data and code in the RAM,” as claimed.

iii. Petitioner's Proposed Combination Fails to Disclose or Suggest a POR Sequencer Descriptor that Includes DMA Controller Descriptors, as Claimed

Claim 1 also recites, in relevant part, “wherein the POR sequencer descriptor includes ... DMA controller descriptors.” Petitioner incorrectly interprets the claimed “DMA controller descriptors” as Post’s pointers, and overlooks that Kao’s DMA engine does not use or need any DMA controller descriptors to be included as part of a POR sequencer descriptor. The Petition fails to establish that these features are present or suggested by Petitioner’s proposed combination of Post, Kao, and Wilson.

Petitioner concedes that Post fails to disclose any DMA controller, but incorrectly (and illogically) alleges that Post discloses a DMA controller descriptor. In its challenge, Petitioner contends that “Post’s next-page pointers ... would be DMA controller descriptors in the asserted combinations.” Pet., 61. Petitioner further argues that these pointers alone satisfy the DMA controller “descriptors” as claimed, where the pointers are used by a DMA controller such as Kao’s DMA engine 120. Pet., 61-65. But Petitioner’s theory misses the point of a “DMA controller descriptor,” and is inconsistent with Petitioner’s own evidence of a “descriptor.”

As disclosed in the '694 Patent, the “DMA controller descriptors 406” are part of the POR Sequencer Descriptor, are identified by DMA descriptors’ control

information 404, and “are loaded by the POR Sequencer to the DMA controller’s configuration register space.” ’694 Pat., 6:26-28. Similarly, the ’694 Patent discloses that the DMA controller descriptors included in the “POR Sequencer descriptor” “are loaded by the POR Sequencer to the DMA controllers of the system.” ’694 Pat., 2:50-52, 2:54-56. In the exemplary illustration of Figure 5, the DMA controller descriptors include a plurality of DMA controller descriptors, DMA controller descriptor 0 to DMA controller descriptor P. See ’694 Pat., 6:40-42.

As also described, the process of “copying of system firmware 211 from the storage device 209 to the RAM 208” is illustrated in Figure 8, and “starts with the POR sequencer 207 reading a DMA controller descriptor 406 from the RAM 208 and then writing the DMA controller descriptor 406 to the configuration space of an available DMA controller 203 in block 801.” ’694 Pat., 7:48-54. “The POR sequencer 207 then activates the DMA controller 203 in block 802, which would then cause the DMA controller 203 to start the transfer of a system firmware fragment 211 from the storage device 209 to the internal RAM 208 via the storage device controller 204 in block 803.” ’694 Pat., 7:54-59. As such, the ’694 Patent describes a DMA controller descriptor as being read by a POR sequencer 207—not a DMA controller—and providing DMA controller register information to activate an associated DMA controller.

Petitioner’s error on this claim term appears to be founded upon its contention that the “’694 patent describes DMA controller descriptors in this same way, as pointers pointing to additional locations in nonvolatile memory” Pet., 62 (citing Ex. 1003, ¶147). While the ’694 Patent discloses that a last DMA controller descriptor “of a POR sequencer descriptor “points to the next POR sequencer descriptor fragment 502,” this is not the entirety of a DMA controller descriptor as disclosed in the ’694 Patent, already discussed above.

Additionally, Petitioner’s flawed argument that a DMA controller descriptor is satisfied by Post’s pointer is also contradicted Petitioner’s own evidence. In Section IV of the Petition, which is Petitioner’s discussion of the technical background, Petitioner includes Section IV.C directed to Direct Memory Access (“DMA”). See Pet., 9-11. In that discussion Petitioner cites to a textbook entitled *Designing Embedded Hardware* to define a “descriptor” as ““a table specifying byte count, source address, destination address, and a pointer to the next descriptor.”” Pet., 10-11 (quoting Ex. 1008, page 21 (page 39 of the pdf)). Petitioner’s assertion that the claimed “DMA controller descriptor” is satisfied solely by Post’s pointer contradicts Petitioner’s own evidence regarding what it contends must be included in a “descriptor.”

Finally, while Petitioner relies on Kao to disclose a DMA engine 120, Petitioner does not and cannot identify any DMA controller descriptor in Kao. See

Pet., 61-65. An independent search confirms that Kao fails to disclose any “descriptor,” much less a “DMA controller descriptor” as claimed. Moreover, Kao has no need for any DMA controller descriptor because in Kao, a NAND programmer 302 “burn[s] boot codes 132 (step 304), controller codes 134 (step 306), and OS codes 136 (step 308) into the NAND flash memory 130” before booting, and “utilize[es] the DMA engine located on the volatile memory chip to load boot and controller codes.” Kao, Fig. 3A (“BEFORE BOOTING”), 4:53-58, 3:3-4. This arrangement obviates any need for a DMA controller descriptor in Kao, and is the very characteristic of Kao that yields Kao’s alleged reduced complexity and cost of the host processor. See Kao, 2:64-3:9. Indeed, Kao’s purpose is to develop “an improved system and method for booting an embedded system” that does not include “initializing and managing DMA transfers ... as part of the boot sequence.” Kao, 2:5-15.

For all these reasons, whether considered alone or in Petitioner’s combination, the asserted references of Post, Kao, and Wilson fail to disclose or suggest at least “wherein the POR sequencer descriptor includes ... DMA controller descriptors,” as claimed.

iv. Ground 1’s Challenge Fails to Establish the Unpatentability of Claim 1

For all of the reasons presented above, Petitioner’s proposed combination of Post, Kao, and Wilson, as combined, fails to disclose or suggest every claimed

feature of claim 1. Therefore, claim 1 survives Petitioner’s Ground 1 challenge. Further, the remaining challenged claims either depend from claim 1 or recite features substantially similar to claim 1. *See, e.g.*, Pet., 69-71 (relying on the analysis of claim 1 to challenge claim 6’s “apparatus” and claim 12’s “article of manufacture”). Therefore, for the same reasons already present in this Section III, all challenged claims, including independent claims 6 and 12, along with dependent claims 5 and 9, survive Petitioner’s Ground 1 challenge.

B. Ground 2 based on Post, Kao, Wilson, and So Also Fails

Petitioner’s addition of So to the Ground 1 combination of Post, Kao, and Wilson is provided to allegedly disclose the claimed “reset controller.” However, this addition of So is not provided to remedy the above-identified shortcomings of the Post-Kao-Wilson combination. *See* Pet., 25-27 (“In the resulting combination of Post, Kao, Wilson, and So, Post’s SoC 110 would be further modified to include So’s reset controller.”). Each of the shortcomings identified above also apply to Petitioner’s challenge based on the asserted combination of Post, Kao, Wilson, and So.

As such, for the same reasons presented above in Section III.A, the challenged claims also survive the asserted combination of Post, Kao, Wilson, and So of Ground 2.

IV. CONCLUSION

For the reasons presented above, each Ground is deficient, and no *inter partes* review should be instituted.

Dated: August 15, 2023

/s/ Wayne M. Helge

Wayne M. Helge (Reg. No. 56,905)
NOSSAMAN, LLP
1401 New York Ave, NW
Suite 800
Washington, DC 20005
Telephone: (202) 887-1400
Email: whelge@nossaman.com

James T. Wilson (Reg. No. 41,439)
DAVIDSON BERQUIST JACKSON &
GOWDEY, LLP
8300 Greensboro Dr., Suite 500
McLean, VA 22102
Telephone: (571) 765-7700
Fax: (571) 765-7200
Email: jwilson@dbjg.com

Counsel for Patent Owner

CERTIFICATE OF WORD COUNT

The undersigned certifies that the foregoing PATENT OWNER'S PRELIMINARY RESPONSE complies with the type-volume limitation in 37 C.F.R. § 42.24(b)(1). According to the word-processing system's word count, the brief contains 4,735 words, excluding the parts of the brief exempted by 37 C.F.R. § 42.24(a).

By: /s/ Wayne M. Helge
Wayne M. Helge (Reg. No. 56,905)
Attorney for Patent Owner

CERTIFICATE OF SERVICE

I hereby certify that on August 15, 2023, a true and correct copy of the foregoing document was served via email, by consent, to Petitioner by serving the correspondence email addresses of record as follows:

Christina J. McCullough
Reg. No. 58,720
PERKINS COIE LLP
1201 Third Avenue
Suite 4900
Seattle, Washington 98101-3099
(206) 359-6897
mccullough-ptab@perkinscoie.com
Intel-BitMicro-
084IPRService@perkinscoie.com

Lori A. Gordon
Reg. No. 50,633
PERKINS COIE LLP
700 Thirteenth Street, N.W.,
Suite 800
Washington, DC 20005-3960
(202) 654-6200
gordon-ptab@perkinscoie.com

Christopher Kelley
Reg. No. 42,714
PERKINS COIE LLP
3150 Porter Drive
Palo Alto, CA 94304-1212
(650) 838-4300
kelley-ptab@perkinscoie.com

Chad S. Campbell
PERKINS COIE LLP
2901 N. Central Avenue
Suite 2000
Phoenix, AZ 85012
campbell-ptab@perkinscoie.com

/s/ Wayne M. Helge
Wayne M. Helge (Reg. No. 56,905)
Attorney for Patent Owner