

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MEDIATEK, INC
Petitioner,

v.

DAEDALUS PRIME LLC,
Patent Owner.

Case No.: IPR2025-00243
Patent 8,769,316

**DECLARATION OF MURALI ANNAVARAM, PH.D.
IN SUPPORT OF PETITION FOR INTER PARTES REVIEW OF
U.S. PATENT NO. 8,769,316**

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	QUALIFICATIONS AND BACKGROUND	2
III.	MATERIALS CONSIDERED	7
IV.	LEGAL STANDARDS	9
A.	Level of Ordinary Skill	10
B.	Prior Art.....	11
C.	Anticipation	11
D.	Obviousness.....	12
V.	THE '316 PATENT	17
VI.	LEVEL OF ORDINARY SKILL IN THE ART	21
VII.	CLAIM CONSTRUCTION	22
VIII.	THE PRIOR ART	23
A.	Finkelstein	23
B.	Felter.....	27
C.	Therein.....	29
IX.	GROUND OF UNPATENTABILITY.....	32
A.	Ground 1: Claims 8 and 12 Would Have Been Obvious Based on Felter.....	33
1.	Independent claim 8	33
a.	[8pre]: “A method comprising:”	33
b.	[8a]: “determining, in a power controller of a multi-domain processor, a power budget for the multi-domain processor for a current time interval, the multi-domain processor including at least a first domain and a second domain”	34

c.	[8b]: “determining, in the power controller, a portion of the power budget to be allocated to the first and second domains, including”	38
d.	[8c]: “allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain, and”	40
e.	[8d]: “sharing a remaining portion of the power budget according to a first sharing policy value for the first domain and a second sharing policy value for the second domain; and”	42
f.	[8e]: “controlling a frequency of the first domain and a frequency of the second domain based on the allocated portions.”	44
2.	Claim 12: “ The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”	46
B.	Ground 2: Claims 13, 14, and 17 Would Have Been Obvious Based on Finkelstein	50
1.	Independent Claim 13	50
a.	[13pre]: “a system comprising:”	50
b.	[13a]: “a multicore processor having a first domain including a plurality of cores, a second domain including a graphics engine, and a third domain including system agent circuitry,”	52
c.	[13b]: “the third domain to operate at a fixed power budget and to dynamically allocate a variable power budget between the first and second domains, wherein”	61

d.	[13c]: “the system agent circuitry includes a power sharing logic to determine the variable power budget for a current time interval and”	66
e.	[13d]: “to allocate a first portion of the variable power budget to the first domain according to a first power sharing value for the first domain, and to allocate a second portion of the variable power budget to the second domain according to a second power sharing value for the second domain; and”	67
f.	[13e]: “a dynamic random access memory (DRAM) coupled to the multicore processor.”	69
2.	Claim 14: “The system of claim 13, wherein the power sharing logic is to dynamically allocate substantially all of the variable power budget to the first domain for a first workload, and to dynamically allocate substantially all of the variable power budget to the second domain for a second workload executed after the first workload.”	70
3.	Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.”	79
C.	Ground 3: Claims 8 and 12-17 Would Have Been Obvious over Finkelstein and Felter	84
1.	Overview of the Finkelstein-Felter Combination	84
2.	A POSITA Would Have Found It Obvious to Adapt Finkelstein’s Power Management Logic to Perform Felter’s “Modified Pro Rata Technique” for Power Allocation.....	88
3.	Independent claim 8	94
a.	[8pre]: “A method comprising:”	94
b.	[8a]: “determining, in a power controller of a multi-domain processor, a power budget for the	

	multi-domain processor for a current time interval, the multi-domain processor including at least a first domain and a second domain”	96
c.	[8b]: “determining, in the power controller, a portion of the power budget to be allocated to the first and second domains,”	101
d.	[8c]: “including allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain, ”	104
e.	[8d]: “and sharing a remaining portion of the power budget according to a first sharing policy value for the first domain and a second sharing policy value for the second domain; and”	107
f.	[8e]: “controlling a frequency of the first domain and a frequency of the second domain based on the allocated portions.”	110
4.	Claim 12: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”	112
5.	Independent Claim 13	116
a.	[13pre]: “A system comprising:”	116
b.	[13a]: “a multicore processor having a first domain including a plurality of cores, a second domain including a graphics engine, and a third domain including system agent circuitry,”	118
c.	[13b]: “the third domain to operate at a fixed power budget and to dynamically allocate a variable power budget between the first and second domains, wherein”	119

- d. [13c]: “the system agent circuitry includes a power sharing logic to determine the variable power budget for a current time interval and” 122
- e. [13d]: “to allocate a first portion of the variable power budget to the first domain according to a first power sharing value for the first domain, and to allocate a second portion of the variable power budget to the second domain according to a second power sharing value for the second domain; and” 124
- f. [13e]: “a dynamic random access memory (DRAM) coupled to the multicore processor.” 125
- 6. Claim 14: “The system of claim 13, wherein the power sharing logic is to dynamically allocate substantially all of the variable power budget to the first domain for a first workload, and to dynamically allocate substantially all of the variable power budget to the second domain for a second workload executed after the first workload.” 125
- 7. Claim 15: “The system of claim 13, wherein the power sharing logic is to increment the first power sharing value when a request for a higher frequency for the first domain is not granted, and to increment the second power sharing value when a request for a higher frequency for the second domain is not granted.” 135
- 8. Claim 16: “The system of claim 13, wherein the power sharing logic is to further allocate the first portion of the variable power budget according to a first minimum reservation value for the first domain and allocate the second portion of the variable power budget according to a second minimum reservation value for the second domain.” 136
- 9. Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.” 137

D.	Ground 4: The Finkelstein-Felter-Therien Combination Renders Obvious Claims 12, 15, and 17	139
1.	Claim 15: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”	139
2.	Claim 12: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”	144
3.	Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.”	147
E.	Ground 5: The Felter-Therien Combination Renders Obvious Claim 12	151
1.	Claim 12: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”	151
F.	Ground 6: The Finkelstein-Therien Combination Renders Obvious Claims 15 and 17	153
1.	Claim 15: “The system of claim 13, wherein the power sharing logic is to increment the first power sharing value when a request for a higher frequency for the first domain is not granted, and to increment the second power sharing value when a request for a higher frequency for the second domain is not granted.”	153

2. Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.”.....155

X. CONCLUSION.....156

I. INTRODUCTION

1. I, Murali Annavaram, submit this declaration to state my opinions on the matters described below.

2. I have been retained by MediaTek, Inc. (“Petitioner”) as an independent expert consultant in this proceeding before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I am being compensated at my usual and customary rate of \$650 USD per hour. No part of my compensation depends on the outcome of this proceeding, and I have no other interest in this proceeding.

3. I am over 21 years of age and, if I am called upon to do so, I would be willing and able to testify as to the matters set forth herein.

4. I understand that this proceeding involves U.S. Patent No. 8,769,316 (“the ’316 Patent”), which I also understand is being submitted as Exhibit 1001 in an accompanying petition for *inter partes* review. I have included a table in Section III below that relates the documents I discuss in this declaration to what I understand the exhibit numbers in the petition to be.

5. I have been asked by Petitioner to consider the validity of certain claims of the ’316 patent based on certain prior art references. I have also been asked to consider the state of the art and prior art available as of September 6, 2011, which is the earliest claimed priority date on the face of the ’316 patent. Based on the combination of prior art references discussed in this declaration, it is my opinion that

claims 8 and 12-17 of the '316 patent are unpatentable for at least the reasons I provide below.

6. Throughout this declaration, I refer to specific pages, figures, or line numbers of various exhibits. These citations are illustrative and are not intended to suggest that they are the only support for the propositions for which they are cited.

II. QUALIFICATIONS AND BACKGROUND

7. I believe that I am well qualified to serve as a technical expert in this matter based upon my qualifications. I have furnished MediaTek's legal representatives with a copy of my curriculum vitae, which I have been told is included as Exhibit 1004. It provides a comprehensive description of my academic and employment history. I am also competent to write this declaration. In formulating my opinions, I have relied upon my training, knowledge, and experience in the relevant art, as described below.

8. I am an expert in the field of computer architecture, including microelectronics architecture design. My research focuses on efficient power management of chip multiprocessors (CMPs) and graphics processing units (GPUs), power efficient heterogeneous CMP designs, system-level power and performance tradeoffs, and reliability of computing platforms, superconducting CPU designs and efficient machine learning systems.

9. I am currently the Lloyd Hunt Chair professor in the Ming-Hsieh Department of Electrical and Computer Engineering, with a joint appointment in the Thomas Lord Computer Science department at the University of Southern California. I held the Dean's Chair Professorship from 2018-2021. I held the position of Robert G. and Mary G. Lane Early Career Chair at USC from 2012 to 2017. I started as a faculty member at USC in 2007.

10. I am founding and inaugural director of the USC-Meta center for Research and Education in AI and Learning at USC from 2022. I am the architecture thrust leader for the NSF's Expeditions in Computing center at USC focused on energy efficiency CPUs using superconducting technologies.

11. I co-authored Parallel Computer Organization and Design, a widely used textbook for graduate-level computer architecture courses. I teach several undergraduate and graduate computer architecture courses at USC, including topics such GPUs, out of order processor design, power and energy efficient design of computer hardware, mobile and edge computing systems, and memory system design.

12. I received a Bachelor of Technology degree in Computer Science from the National Institute of Technology in Warangal, India, in 1993 and a Master of Science degree in Computer Science and Engineering from Colorado State University in 1996. I received my Ph.D. degree in Computer Science and

Engineering from The University of Michigan in 2001. I worked as a Senior Research Scientist at Intel Corporation from 2001-2007 and as a Visiting Faculty Researcher at Nokia Research Center in 2007 before joining the faculty of the University of Southern California in August 2007. I was a visiting faculty scientist at Meta during 2020-2021. During 2022-2023 I held the Rukmini Gopalakrishnachar visiting chair professorship at the Indian Institute of Sciences, Bengaluru, India.

13. I have received numerous awards, including, for example, the IBM Faculty Partnership award in 2008, the Best Paper Award at the 2009 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS) and the National Science Foundation CAREER award in 2010, Best Paper Award at the Federated Machine Learning workshop, and Best paper and Best Student paper finalist nominations at the Supercomputing 2019 conference. I am a Fellow of the Institute of Electrical and Electronics Engineers (IEEE) and a distinguished member of the Association of Computing Machinery (ACM).

14. I have had dozens of papers accepted on the topic of power efficient hardware design at major conferences in the field of computer architecture, including the International Symposium on Computer Architecture (ISCA), the High Performance Computer Architecture (HPCA), and the Symposium on Microarchitecture (MICRO).

15. My research on server energy efficiency has been selected for the IEEE Micro TopPicks award for the most influential computer architecture paper of the year 2013.

16. I have given keynote presentations at the Frontiers in Technology distinguished speaker series at the University of the California, Merced, 2021 Annual KIXX forum distinguished speaker series at the King Fahd university, Swedish Multicore Symposium in 2018 and at the IEEE International On-Line Test Symposium in 2014.

17. For my research and publications in this field, I was inducted into the ACM Special Interest Group on Microarchitecture Hall of Fame in December 2015. I am only one of 47 researchers over the past 48 years to receive this honor. I was also inducted into the IEEE Computer Society Technical Committee on Computer Architecture's High Performance Computer Architecture Hall of Fame. I am only one of 42 researchers over the past 22 years to receive this honor. Finally, I was inducted into the ACM International Symposium on Computer Architecture (ISCA) Hall of Fame in June 2017. I am only one of 81 researchers to have received this honor over the past 44 years at the time of award selection.

18. I served as the General Co-Chair for ISCA 2018 and the Technical Program Chair for HPCA 2021, International Conference on Supercomputing 2024, Architecture Track Program Chair for HiPC 2014. I have also served as a program

committee member for several international conferences and symposiums, including ISCA, MICRO, HPCA, SIGMETRICS, DSN, HiPC, and ASPLOS.

19. I served as an associate editor for the technical journal of the Association for Computer Machinery Transactions on Design Automation of Electronic Systems (ACM TODAES), and I served as a journal reviewer for the ACM Transactions on Computer Systems (ACM TOCS), ACM Transactions on Embedded Computing Systems (ACM TECS), ACM Computing Surveys, ACM Transactions on Architecture and Code Optimization (ACM TACO), and IEEE Top Picks in Microarchitecture 2010.

20. I am an inventor or co-inventor of seven issued patents relating to power and reliability of computing systems.

21. I graduated 16 PhD students at USC, and 13 of them were solely advised by me. Of these, 6 are now in academia (both in the USA and abroad) as professors working on various computer architecture related problems.

22. My research group has published about 30 papers on the topic of multi-core CPUs and GPUs with power optimized designs at top-tier computer architecture conferences in the past 10 years, where the typical acceptance rates are less than 20%. In particular, my group published 3 papers at the flagship computer architecture conference ISCA during 2016, which is a record for the highest number of published papers by a single research group at that time. These papers focused on

improving the power and performance tradeoffs for various power efficiency techniques applied to highly parallel and heterogeneous processors such as CMPs, GPUs.

23. My research group has designed and evaluated the energy efficient techniques for heterogeneous CMPs, including building fully functional prototypes that required substantial understanding of the interactions between microarchitecture, circuits, operating systems and compilers. For instance, my group's work on KnightShift energy proportional computing developed an extensive set of power management schemes at chip and system level for heterogeneous computing platforms in active low-power state.

24. In my group we routinely work with various benchmarks, including SPEC, ParSec, to measure the power and performance impacts of architecture enhancements to chip designs. We also use various profiling tools, such as PIN and Vtune, for measuring various chip and system level activities. We use detailed microarchitecture simulation tools, such as SimpleScalar, GPGPUsim, for power and performance evaluations.

III. MATERIALS CONSIDERED

25. In forming my opinions, I have reviewed the following documents, as well as other documents cited throughout this declaration:

Exhibit	Description
1001	U.S. Patent No. 8,769,316 to Ananthakrishnan et al. (the “’316 patent”).
1002	Prosecution History of the ’316 patent (“Prosecution History”).
1005	U.S. Patent Appl. Pub. No. 2010/0115304 to Finkelstein et al (“Finkelstein”).
1006	U.S. Patent Appl. Pub. No. 2006/0288241 to Felter et al. (“Felter”).
1007	U.S. Patent Appl. Pub. No 2010/0162006 to Therien et al (“Therien”).
1008	Linley Gwennap, <i>Sandy Bridge Spans Generations: Intel Focuses on Graphics, Multimedia in New Processor Design</i> , Microprocessor Report, September 2010 (“Gwennap”).
1009	U.S. Patent Appl. Pub. No 2009/0328055 to Bose et al. (“Bose”).
1010	U.S. Patent Appl. Pub. No. 2008/0288748 to Sutardja et al (“Sutardja”).
1011	U.S. Patent Appl. Pub. No. 20070049133 to Conroy et al (“Conroy”).
1012	U.S. Patent Appl. Pub. No. 2009/0322768 to Lalonde et al. (“Lalonde”).

26. I have also relied on my education, experience, research, training, and routine knowledge in the relevant art, as well as my understanding of certain legal principles described in this declaration.

27. All of my opinions contained in this declaration are based on the documents I reviewed and my knowledge and professional judgment. My opinions have also been guided by my understanding of how a POSITA would have

understood the claims of the '316 Patent at the time of the earliest claimed priority date.

28. I reserve the right to supplement and amend any of my opinions in this declaration based on documents, testimony, and other information that becomes available to me after the date of this declaration.

IV. LEGAL STANDARDS

29. I am an engineer and not a lawyer. My understanding of the legal standards to apply in reaching the conclusions in this declaration is based on discussions with counsel for Petitioner, my experience applying similar standards in other patent-related matters, and my reading of the documents submitted in this proceeding. I have applied these legal standards in preparing this declaration.

30. I have been informed about two ways in which prior art may render a patent claim unpatentable. First, I have been informed that the prior art can “anticipate” a claim. Second, I have been informed that the prior art can render a claim “obvious” to a POSITA. I understand that a claim is patentable if it was not anticipated and would not have been rendered obvious by the prior art at the effective filing date of the patent.

31. I have been informed that a dependent claim is a patent claim that refers back to another patent claim. I have been informed that a dependent claim includes

all of the limitations of the claim to which it refers plus its own additional limitation(s).

32. I have been asked to provide my opinions as to whether the cited prior art discloses or renders obvious claims 8 and 12-17 of the '316 Patent from the perspective of a POSITA as of September 6, 2011, as I describe in more detail below.

33. I have been informed that in IPR proceedings, such as this one, the party challenging the patent bears the burden of proving unpatentability by a preponderance of the evidence. I understand that a preponderance of the evidence means "more likely than not."

34. For purposes of this declaration, I have been asked to provide my opinions on issues regarding unpatentability. I have been informed of the following legal standards, which I have applied in forming my opinions.

A. Level of Ordinary Skill

35. I have been informed that a POSITA is determined by considering several factors, including the (i) type of problems encountered in the art; (ii) prior art solutions to those problems; (iii) rapidity with which innovations are made; (iv) sophistication of the technology; and (v) educational level of active workers in the field.

36. I have been instructed to assume that a POSITA is not a specific real individual, but rather a hypothetical individual having the qualities reflected by the

factors discussed above. I have also been instructed that a POSITA is assumed to be person of ordinary creativity familiar with the prior art as of the priority date of the patent at issue.

B. Prior Art

37. I have been advised and understand that the information used to evaluate whether an invention was new and not obvious when made is generally referred to as “prior art.” I understand that in an IPR proceeding, prior art includes patents and printed publications that existed before the earliest claimed priority date or the earliest filing date of the patent (which I have been informed is also called the “effective filing date”). I have been informed and understand that a patent or published patent application is prior art if it was filed before the earliest filing date of the claimed invention and that a printed publication is prior art if it was publicly available before the earliest filing date.

C. Anticipation

38. I have been informed that under 35 U.S.C. § 102, a patent claim is unpatentable for anticipation if the claimed subject matter was patented or described in a printed publication before the effective filing date of the claimed invention. I have been informed that this is referred to as unpatentability by anticipation. I have been informed that a patent claim is anticipated under § 102 if a single prior art reference discloses all the limitations of the claimed invention. I understand that

limitations may be expressed or inherent such that the limitation is essential to the prior art.

D. Obviousness

39. I have been informed that a patent claim is unpatentable for obviousness under 35 U.S.C. § 103 if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious to a POSITA to which said subject matter pertains at the time the invention was made. I have been informed that this is referred to as unpatentability by obviousness.

40. When considering obviousness, I have been told that I am to do the following:

- a. Determine the scope and content of the prior art;
- b. Ascertain the differences between the prior art and the claims at issue;
- c. Resolve the level of ordinary skill in the pertinent art; and
- d. Consider evidence of secondary indicia of nonobviousness (if available).

41. I have been informed that the relevant time for considering whether a claim would have been obvious to a POSITA is the time of invention. For my obviousness analysis, counsel for Petitioner instructed me to assume that the date of

invention for the challenged claims is **September 6, 2011**. My opinions would not change if I assumed another, e.g., later, date of invention.

42. I have been informed that a reference may be modified or combined with other references or with a POSITA's own knowledge if the person would have found the modification or combination obvious. I have also been informed that a POSITA is presumed to know all the relevant prior art, and the obviousness analysis may take into account the inferences and creative steps that a POSITA would employ.

43. I have been informed that an obviousness determination must be made from the perspective of a POSITA. I have also been informed that there is no requirement that the prior art contain an express suggestion to combine known elements to achieve the claimed invention, and that a suggestion to combine known elements to achieve the claimed invention may come from the prior art as a whole or individually. Also, the obviousness analysis may rely on the inferences and creative steps a POSITA would employ, as filtered through his or her knowledge as of the priority date. But I understand that obviousness grounds cannot be sustained by mere conclusory statements and must include some articulated reasoning and rationale to support a legal conclusion of obviousness.

44. In determining whether a prior art reference could have been combined with another prior art reference or other information known to a POSITA, I have

been informed that the following principles may be considered, and I have been asked to consider them:

- a. A combination of familiar elements according to known methods is likely to be obvious if it yields predictable results;
- b. The substitution of one known element for another is likely to be obvious if it yields predictable results;
- c. The use of a known technique to improve similar items or methods in the same way is likely to be obvious if it yields predictable results;
- d. The application of a known technique to a prior art reference that is ready for improvement to yield predictable results;
- e. Any need or problem known in the field and addressed by the reference can provide a reason for combining the elements in the manner claimed;
- f. A person of ordinary skill often will be able to fit the teachings of multiple references together like a puzzle; and
- g. The proper analysis of obviousness requires a determination of whether a POSITA would have a “reasonable expectation of success”—but not “absolute predictability” of success—in achieving the claimed invention by combining prior art references.

45. I have been informed that, when a work is available in one field, design alternatives and other market forces can prompt variations of it, either in the same field or in another. I have been informed that if a POSITA could have implemented a predictable variation and would have seen the benefit of doing so, that variation is likely to have been obvious. I have been informed that, in many fields, such as the mechanical or electrical arts, market demand—not scientific literature—may drive

design trends. I have been informed that, when there was a design need or market pressure and there are a finite number of predictable solutions, a POSITA would have had a good reason to pursue those known options.

46. I have been told that a claimed invention may be obvious even when the prior art does not teach each claim limitation, so long as the record contains some reason why a person of ordinary skill in the art would have applied the prior art to obtain the claimed invention. I have been informed that the law permits the application of “common sense” in examining whether a claimed invention would have been obvious to a POSITA. For example, I have been informed that combining familiar elements according to known methods and in a predictable way may suggest obviousness when such a combination would yield nothing more than predictable results. I understand, however, that a claim is not obvious merely because every claim element is disclosed in the prior art. A party asserting obviousness must provide a specific motivation to combine or modify the references as recited in the claims and explain why one skilled in the art would have reasonably expected to succeed in doing so.

47. I have been informed that there is no rigid rule that a reference or combination of references must contain a “teaching, suggestion, or motivation” to combine references. But I also understand that the “teaching, suggestion, or motivation” test can be a useful guide in establishing a rationale for combining

elements of the prior art. I have been informed that this test poses the question as to whether there is an express or implied teaching, suggestion, or motivation to combine prior art elements in a way that results in the claimed invention, and that it helps to counter the use of hindsight, which is impermissible. Likewise, if a prior art reference “teaches away” from a potential prior art combination, then a motivation to combine may not exist. I have been told the reasons a POSITA would have wanted to modify teachings available in the prior art teachings do not need to match the inventor’s reasons for making the claimed invention. In determining whether the subject matter of a patent claim is obvious, I have been told that what matters is the objective reach of the claim and whether it covers an obvious variation of the prior art. I have been told it would be improper to consider only the specific problem the patentee was trying to solve.

48. I am not aware of any evidence of secondary considerations, such as unexpected results, industry skepticism, long-felt unresolved need, commercial success, praise by others, or copying that would alter my opinions set forth below.

49. I have been informed that, in an obviousness analysis, prior art must be analogous art to the patent being considered. I have been informed that a prior art reference is considered to be analogous, or in the same field of art, if the reference is either (1) in the same field of endeavor as the challenged patent, regardless of the

problems the challenged patent and the prior art address, or (2) reasonably pertinent to the particular problem being solved by the challenged patent.

V. THE '316 PATENT

50. I understand that the '316 patent was filed on September 6, 2011, and issued on July 1, 2014. '316 patent, cover page. I have been asked to treat **September 6, 2011**, as the date from which to assess my opinions regarding patentability and the knowledge a POSITA would have possessed.

51. The '316 patent is titled “Dynamically Allocating a Power Budget Over Multiple Domains of a Processor.” As its title suggests, the '316 patent generally describes ways to budget power for a multidomain processor and explains how to allocate that power budget “between the different domains of a multicore processor.” '316 patent, 2:10-16. Figure 1, which I have reproduced below, is “a flow diagram of a high level method of performing power budget allocations between multiple domains.” '316 patent, 5:11-14. This flow diagram, which would have been typical for budgeting and allocating power in a multidomain processor, has three high-level steps: “determining a power budget for a multi-domain processor for a current time interval, determining a portion of the power budget to be allocated to first and second domains of the processor, and controlling a frequency of the domains based on the allocated portions.” '316 patent, Abstract; *see also* '316 patent, 5:11-43.

100

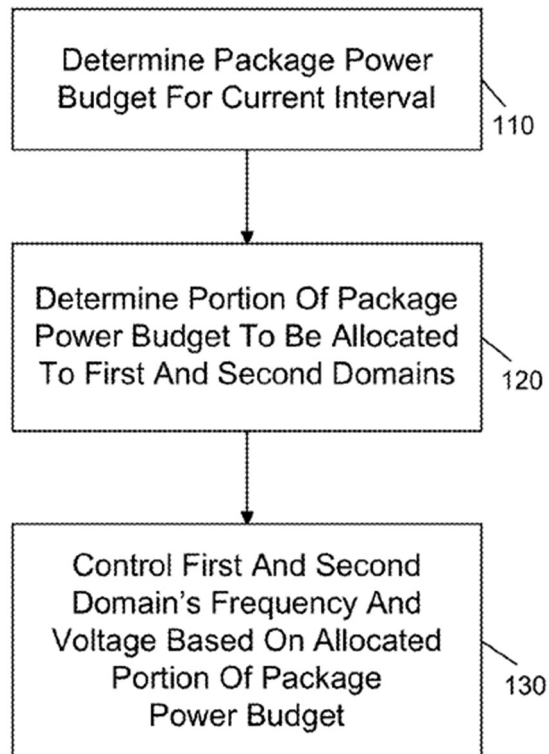


FIG. 1

'316 Patent, FIG. 1.

Although called a “power budget,” the '316 patent makes clear that the “power budget” may be an energy budget measured in Joules rather than a power budget measured in Watts. For example, “the power budget can be calculated using the following equation: $E_n = E_{n-1} * \alpha + (1 - \alpha) * \text{Power_Limit} * \Delta T - \text{Energy}$) where E_n =energy budget for the current (Nth) evaluation instant (which can be measured

in Joules)” ’316 Patent, 3:22-62. The same equation appears in claim 7, which states that “ E_n is the power budget for the current time interval.” ’316 patent, claim 7. A POSITA would have understood this is because a power budget can be converted to an energy budget, and vice versa, simply by multiplying the power budget by the time interval (or dividing the energy budget by the time interval).

52. In one example, a multicore processor “can include at least two independent domains, one associated with the cores (referred to . . . as a core domain) and one associated with a graphics engine (referred to . . . as a graphics domain).” ’316 patent, 1:59-62. Figure 7 below illustrates such an example. The ’316 patent describes Figure 7 below as follows:

Specifically, a core domain 410 can include a plurality of cores 410a-410n, a graphics domain 420 can include one or more graphics engines, and a system agent domain 450 may further be present. In various embodiments, system agent domain 450 may execute at a fixed frequency and may remain powered on at all times to handle power control events and power management such that domains 410 and 420 can be controlled to dynamically enter into and exit low power states. In addition, these domains can dynamically share a package power budget between them in accordance with an embodiment of the present invention. Each of domains 410 and 420 may operate at different voltage and/or power.

’316 patent, 9:5-17.

400

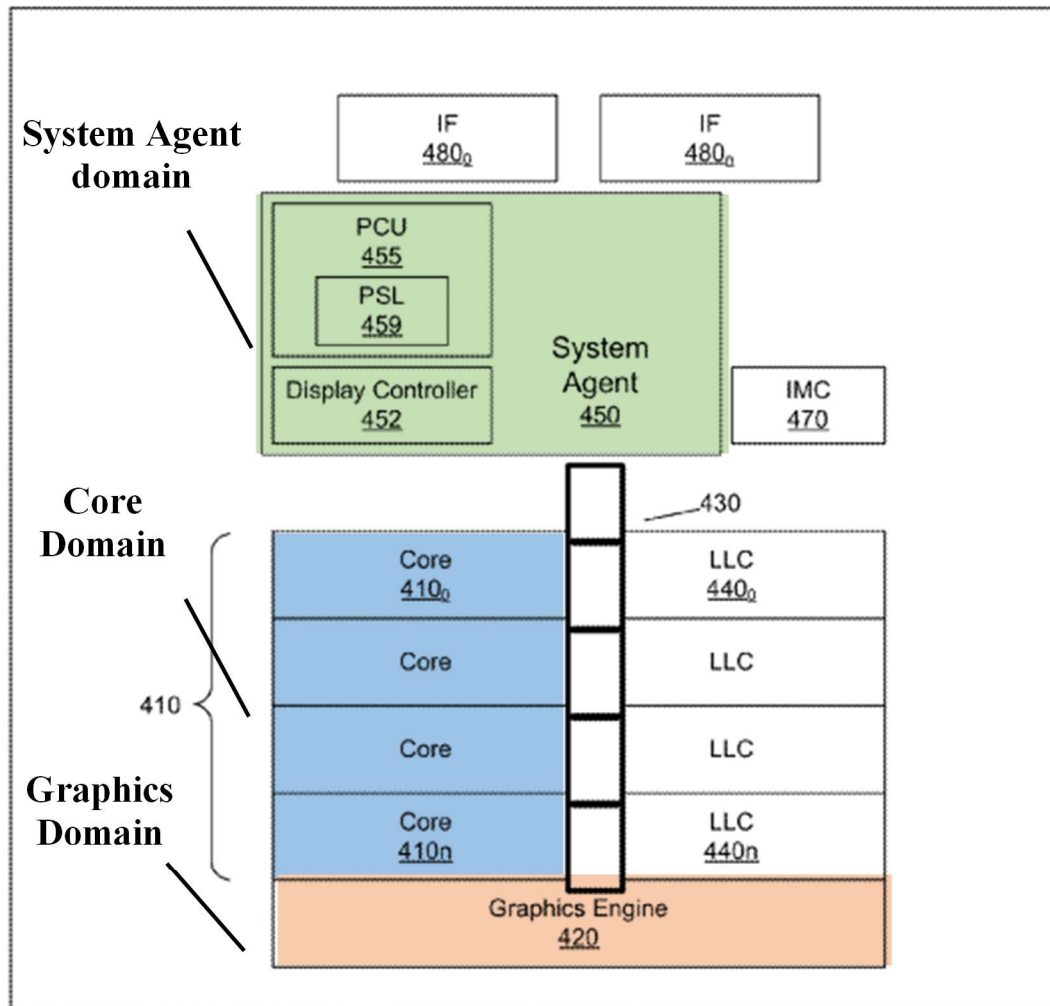


FIG. 7

'316 patent, FIG. 7 (annotated)

53. The '316 patent further explains that “a power budget management (PBM) algorithm may be executed by logic such as logic of a power control unit

(PCU) of a processor to control the power of an entire processor or an individual domain to a configured power limit.” ’316 patent, 2:28-33. For example, “minimum reservation values can be allocated to the domains, and then any remaining package power budget can be shared according to sharing policy values.” ’316 patent, 6:49-53; *see also* ’316 patent, 7:14-27 (explaining the same mathematically via formulas [6] and [7]). “[I]n products having greater than two domains, the analysis may proceed similarly but the available power budget is shared amongst the $m > 2$ variable power domains based on policy values, reservation values, and/or preferences for each of the domains.” ’316 Patent, 6:63-67. Then, based the policy values, “a percentage of the package budget . . . can be allocated to each of the domains by controlling their frequency and/or voltage accordingly.” ’316 patent, 5:3-7. This is an example of dynamic voltage and frequency scaling (“DVFS”), a common technique for power management typically implemented using the “Advanced Configuration and Platform Interface (ACPI) standard.” ’316 patent, 2:48-3:15.

VI. LEVEL OF ORDINARY SKILL IN THE ART

54. In my opinion, a POSITA in the field of the ’316 patent would have had at least a bachelor’s degree in electrical engineering, computer engineering, physics, or a similar field, and approximately two years of industry or academic experience in a field related to microelectronic architecture design and development, including experience with power management in multicore processors. Work experience could

substitute for formal education and additional formal education could substitute for work experience.

55. By September 6, 2011, I had already received a Ph.D. in Computer Engineering from the University of Michigan, Ann Arbor, and had been a professor the Ming-Hsieh Department of Electrical and Computer Engineering at the University of Southern California for four years. I also worked at Intel Microprocessor Research Lab from 2001 to 2007 on energy efficient system design and die-stacking architectures. I would have satisfied the level of ordinary skill in the art, and I am qualified to testify to what one of ordinary skill in the art would have known at that time.

VII. CLAIM CONSTRUCTION

56. I understand that, in this proceeding, the claims are construed according to their ordinary and customary meaning, in light of the specification and prosecution history, as understood by a POSITA at the time of the invention.

57. In my opinion, none of the claim terms require further construction for me to provide my opinions regarding whether claims 8 and 12-17 of the '316 patent are patentable.

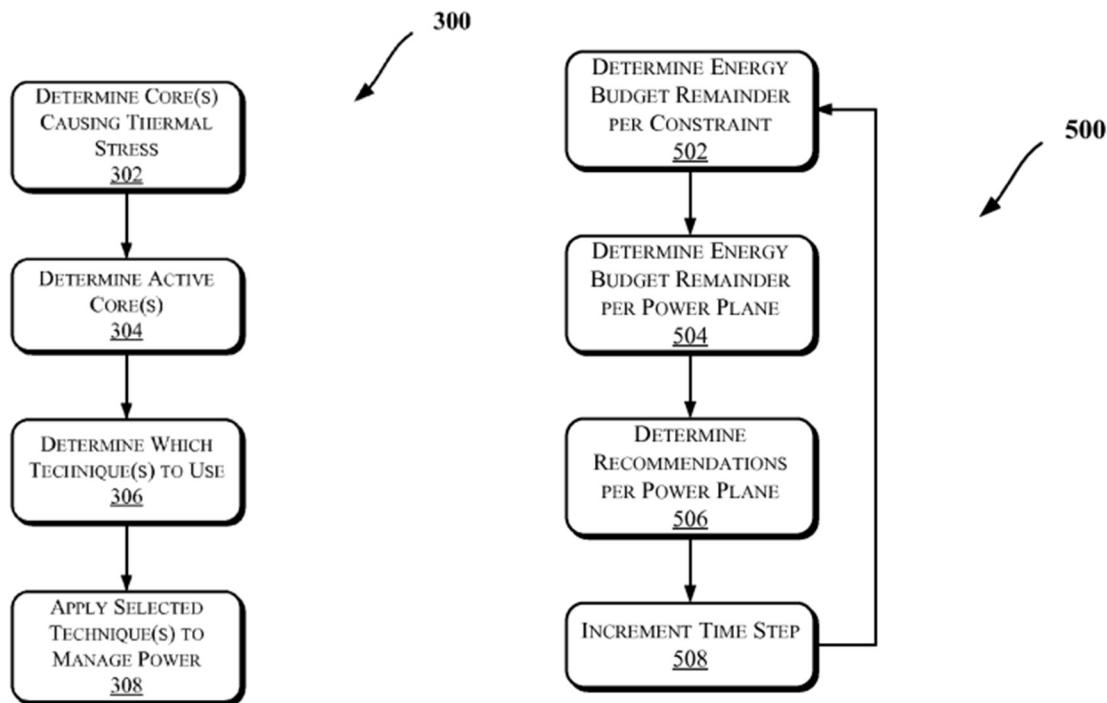
VIII. THE PRIOR ART

58. It is my opinion that claims 8 and 12-17 would have been obvious to a POSITA based on combinations of the Finkelstein, Felter, and Therien references, each of which I have been instructed to assume qualifies as prior art.

A. Finkelstein

59. U.S. Patent Application No. 2010/0115304 A1 to Finkelstein et al. (“Finkelstein”) was filed on October 31, 2008, and published on May 6, 2010. Finkelstein, cover page.

60. Finkelstein is titled “Power Management for Multiple Processor Cores.” It describes, for example, how “power may be distributed among different power planes of a processor based on energy-based considerations.” Finkelstein, abstract. Figures 3 and 5 below are exemplary flow diagrams describing how “to apply one or more selected techniques to multiple cores of a processor to man[a]ge power.” Finkelstein, [0022]. “For example, if multiple power planes sharing the same power source 120 are present in system 100, power management may need to be satisfied for both individual constraints [k] per power plane and global constraints per package.” Finkelstein, [0026].



Finkelstein, FIGS. 3 and 5

As Finkelstein explains in Figure 5 on the right above, “at an operation 502, the energy budget remainder per constraint E_n^k may be determined (e.g., by the logic 140 in accordance with formula (1) . . .).” Finkelstein, [0039]; *see also* Finkelstein, [0028]-[0029] (disclosing formula (1), which defines energy budgets iteratively as: “ $E_{n+1} = \alpha E_n - (TDP_n - P_n) \Delta t_n$ ”). Finkelstein, [0028]-[0029]. Then, “[a]t an operation 504, the energy budget remainder per power plane $E_n^{k,i}$ may be determined (e.g., by the logic 140 in accordance with formula (2) . . .).” Finkelstein, [0039]. More specifically, “[f]or constraint k, user (or application defined) preferences $[W_k]$ that describe how budget E^k is distributed among power planes may be determined.” Finkelstein, [0034]. Power is then allocated among all power planes i that share the

budget E^k according to formula (2): “ $E^{k,i}=W_{ki}E^k$.” Finkelstein, [0034]. Lastly, “the resulting P-state recommendation . . . for power plane i”—an ACPI concept that defines the operating frequency—is assigned using function $f^i(E^{k,i})$. Finkelstein, [0036]; *see also* Finkelstein, [0032]-[0033], [0039], FIG. 4. These are the same three steps from Figure 1 of the ’316 patent. And the process repeats for the next time period. Finkelstein, [0039], FIG. 5 (block 508).

61. Finkelstein’s power management techniques are implemented in a computing system, such as system 100 illustrated in Figure 1. Finkelstein, [0010].

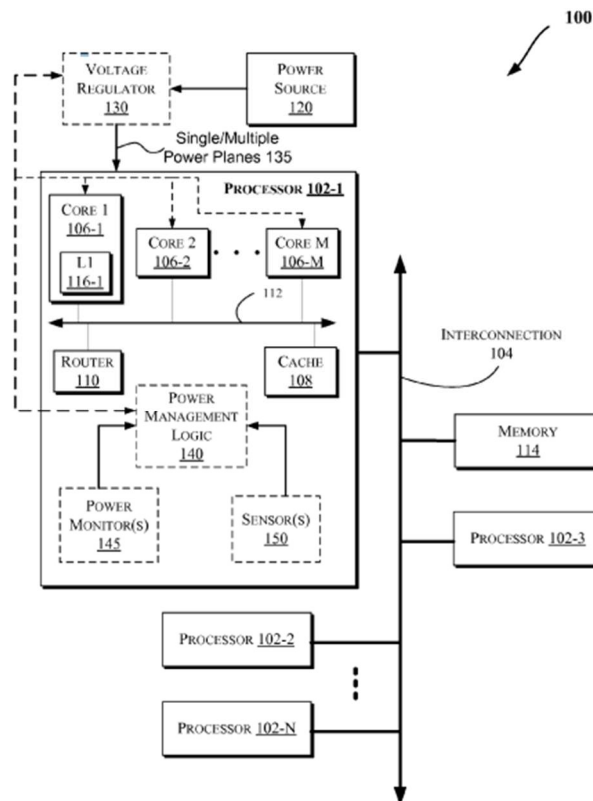


FIG. 1

Finkelstein, FIG. 1

62. As Figure 1 shows above, Finkelstein's computing system 100 includes, among other things, a multicore processor 102-1 having "one or more processor cores 106-1 through 106-M (referred to herein as 'cores 106,' or 'core 106')," power source 120, voltage regulators (VRs) 130, and power management logic 140. Finkelstein, [0011]; *see also* Finkelstein, [0010]-[0016]. Voltage regulator(s) 130 "may be coupled to a single power plane 135 (e.g., supplying power to all the cores 106) or to multiple power planes 135 (e.g., where each power plane may supply power to a different core or group of cores)." Finkelstein, [0014]. Additionally, although not shown explicitly in Figure 1 above, Finkelstein's processor 102-1 may further include a "graphics . . . controller[]," such as a "graphics and memory control hub (GMCH)" with "a graphics interface 614 that communicates with a graphics accelerator 616" over "an accelerated graphics port (AGP)." Finkelstein, [0011], [0044]; *see also* Finkelstein, [0046] ("[T]he processor 602 and the GMCH 608 may be combined to form a single chip. Furthermore, the graphics accelerator 616 may be included within the GMCH 608 in other embodiments of the invention."), FIG. 6.

63. The power management logic 140 in Finkelstein's processor 102-1 "control[s] supply of power" to the cores by "instruct[ing] the VR 130, power source 120, and/or individual components of system 100 (such as the cores 106) to modify their operations." Finkelstein, [0016]. Although the processor cores and

graphics controller in Finkelstein's processor 106-1 may reside on distinct "power planes," they may nevertheless "share a common package power/energy budget." Finkelstein, [0026]. For example, when a power plane for running one or more processor cores shares a package power budget with a power plane for graphics acceleration, "unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload [may be used], when processor resources are not fully utilized." Finkelstein, [0026].

64. Finkelstein is in the same field of endeavor as the '316 patent: power management in microprocessors. *Compare* Finkelstein, [0001], *with* '316 patent, Abstract. Finkelstein is also directed to a problem the '316 patent addresses: ensuring circuitry has sufficient power to perform. *Compare* Finkelstein, [0001]-[0003], [0009] *with* '316 patent, 1:18-22.

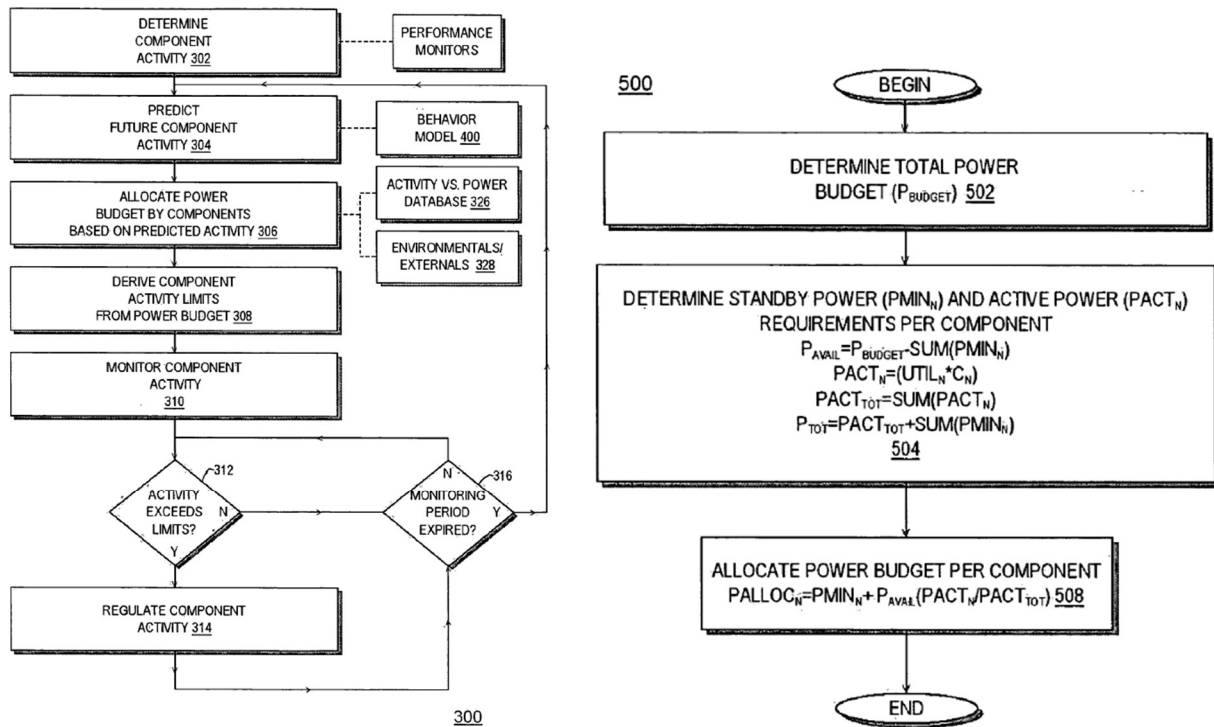
B. Felter

65. U.S. Patent Application No. 2006/0288241 A1 to Felter et al. ("Felter") was filed on June 16, 2005, and published on December 1, 2006. Felter, cover page. Like the '316 patent and Finkelstein, Felter describes "a method for managing power in a data processing system having multiple components." Felter, [0010]. This includes, for example, "single-chip systems in which multiple types of components (e.g., processor cores, cache memories, and the like) are integrated within a single piece of silicon." Felter, [0028]; *see also* Felter, [0037]-[0049] (describing a method

of allocating power to “two independent frequency-scalable processors using a single memory (DRAM) system”). As Felter’s Figures 3 and 5 summarize below, Felter’s power management technique includes the same three steps described in Figure 1 of the ’316 patent: determining a power budget (P_{BUDGET}) (*see* FIG. 5 (block 502)), allocating the power budget to the various system components (*see* FIG. 5 (blocks 504 and 506)), and regulating the operation of the components based on their respective power allocations (*see* FIG. 3 (blocks 306 through 316)), including by DVFS. *See* Felter, [0037] (“Outlined below is an approach to using DVFS”), [0051]-[0066] (describing how to budget and allocate power, as well as how to throttle component activity based on the same), FIGS. 3-5. Felter uses “a modified pro rata technique” for power allocation in which “the amount of power budgeted for each component is equal to the component’s standby power requirements (which must always be met) and a portion of the available [remaining] power P_{AVAIL} .” Felter, [0061], FIG. 5 (blocks 504 and 508). According to Felter, this technique

effectively distributes the available system power to the system components in a manner that (1) ensures that each component receives a minimum level of power and (2) allocates the available active power in proportion to the requirement for the components by the workload(s) executing on the system. The formula expressed in block 508 represents a technique for distributing scarce system power equitably so that the system performance of each component is impacted proportionately.

Felter, [0061].



Felter, FIGS. 3 and 5

66. Felter is in the same field of endeavor as the '316 patent: power management in microprocessors. *Compare* Felter, [0001]-[0002], with '316 patent, Abstract. Felter is also directed to a problem the '316 patent addresses: ensuring circuitry has sufficient power to perform. *Compare* Felter, [0007], [0009] with '316 patent, 1:18-22.

C. Therein

67. U.S. Patent Application Publication No. 2010/0162006 to Therein et al ("Therein") was filed on December 22, 2008, and published on June 24, 2010. Therien, cover page.

68. Therien is titled “Adaptive Power Budget Allocation Between Multiple Components in a Computing System.” It teaches an adaptive “power budget allocation engine” and method for use in a multicomponent (domain) computer system. Therien, Abstract, [0008]. Similar to Finkelstein and Felter, Therien’s computing system 100 includes, among other things, “a first component 110,” “a second component 120,” and “a power budget allocation engine 150” to “facilitate an allocation of an overall power budget among the first and second components 110, 120 as appropriate.” Therein, [0008]-[0010], FIG. 1.

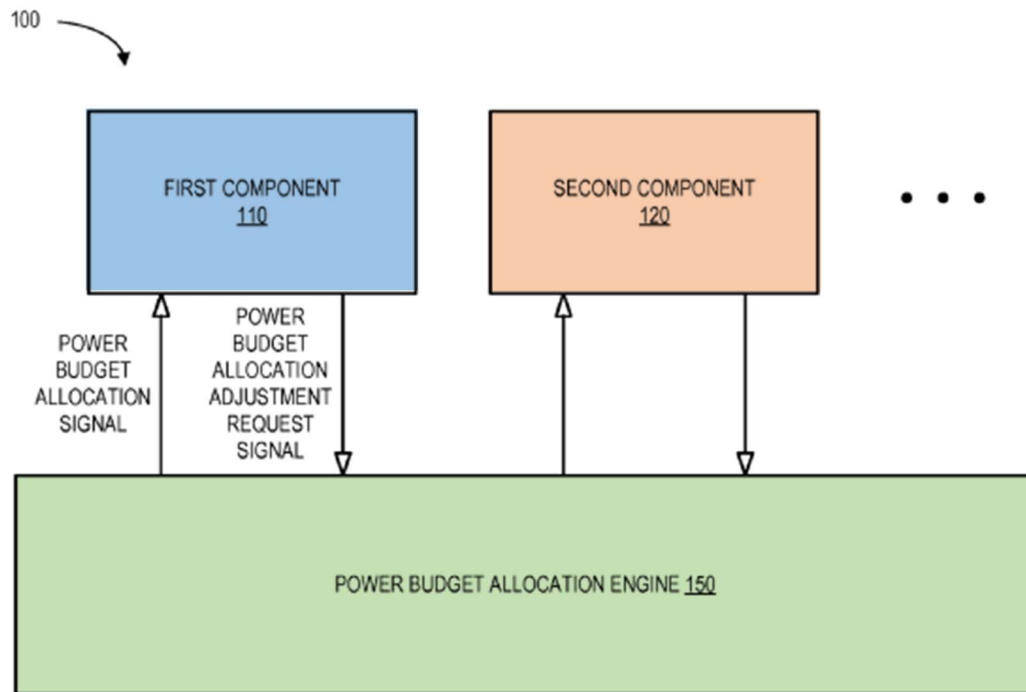
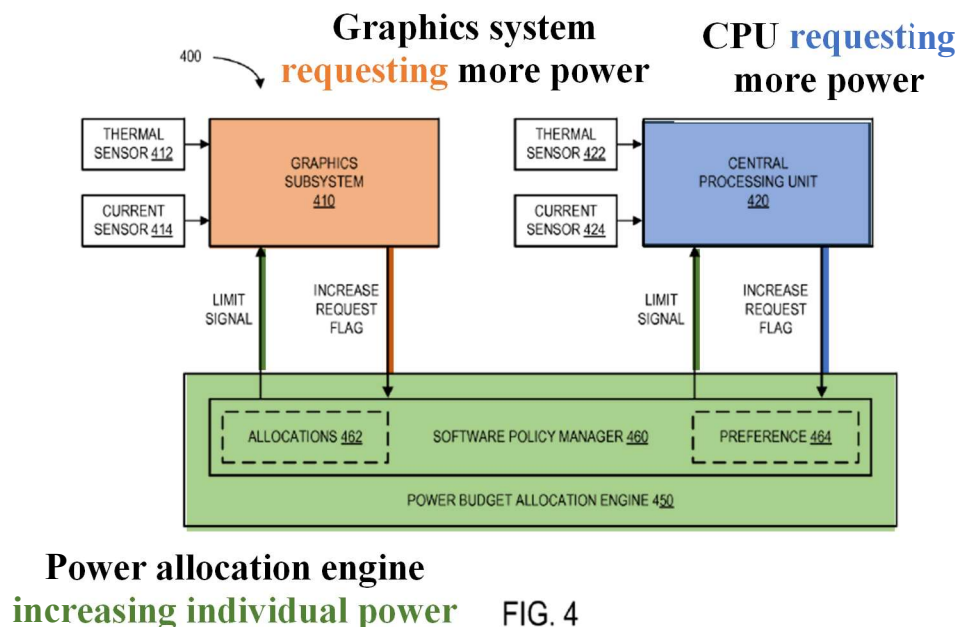


FIG. 1

Therien, FIG. 1 (annotated)

69. Therien's "power budget allocation engine" includes a "policy manager" that "manages a power and/or thermal budget associated with multiple components (and component software interfaces) to facilitate improved results given a system preference." Therien, [0010], [0017], [0026], [0032]-[0034], FIGS. 4-5. To account for increased power demands, CPU and graphics components, for instance, may send a "power budget allocation adjustment request signal," such as an "increase request flag" to the power budget allocation engine. Therien, [0011], [0015]. Upon receipt, the power budget allocation engine may "adjust a power budget allocation signal provided to the . . . component" so that "an individual power limit imposed on a component that may be relaxed or raised as appropriate." Therien, [0013]-[0016], [0026], FIG. 4.



Therien, FIG. 4 (annotated)

70. Therien is in the same field of endeavor as the '316 patent: power management in microprocessors. *Compare* Therien, Abstract, with '316 patent, Abstract. Therien is also directed to a problem the '316 patent addresses: ensuring computing components have sufficient power to perform. *Compare* Therien, Abstract, [0007], [0014]-[0017] with '316 patent, 1:18-22.

IX. GROUNDS OF UNPATENTABILITY

71. Based on my review of the materials listed above in Section III, including my application of the knowledge a POSITA would have possessed, it is my opinion that claims 8 and 12-17 of the '316 patent at least would have been obvious as of September 6, 2011.

72. In particular, it is my opinion that claims 8 and 12-17 of the '316 Patent at least would have been obvious based on the following grounds:

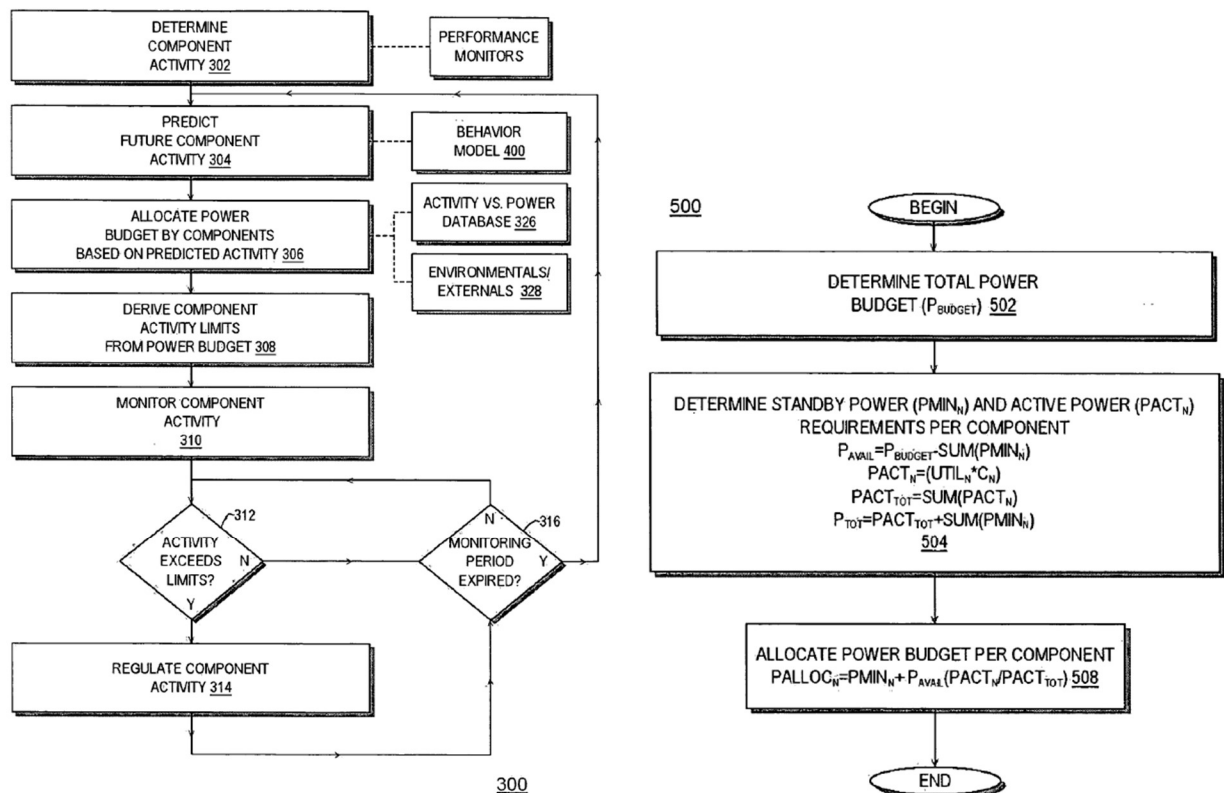
Grounds	Claims	Basis	Prior Art
1	8, 12	§103	Felter
2	13, 14, 17	§103	Finkelstein
3	8, 12-17	§103	Finkelstein, Felter
4	12, 15, 17	§103	Finkelstein, Felter, Therien
5	12	§103	Felter, Therien
6	15, 17	§103	Finkelstein, Therien

A. Ground 1: Claims 8 and 12 Would Have Been Obvious Based on Felter

1. Independent claim 8

a. [8pre]: “A method comprising:”

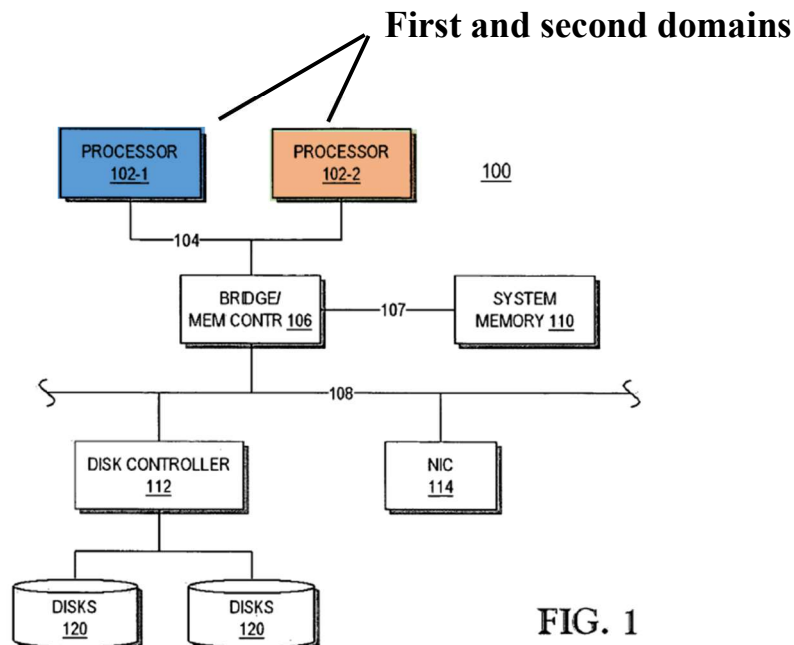
73. Felter discloses feature [8pre] because it discloses a “method for managing power in a data processing system having multiple components.” Felter, Abstract. For example, Felter discloses “a method for allocating power budgets to the system components based on expected levels of activity.” Felter, [0016]; *see also* Felter, [0051]-[0066] (describing “a modified pro rata technique” for power allocation), FIGS. 3, 5.



Felter, FIGS. 3 and 5

- b. [8a]: “determining, in a power controller of a multi-domain processor, a power budget for the multi-domain processor for a current time interval, the multi-domain processor including at least a first domain and a second domain”

74. Felter discloses feature [8a] because it discloses a “data processing system” (multi domain processor) such as a “single-chip system[] in which multiple types of components (e.g., processor cores . . .) are integrated within a single piece of silicon” (multi-domain processor) and a “power administrator” (power controller) that controls “the total power budget (P_{BUDGET}) of the system.” Felter, [0028], [0030], [0057]; *see also* Felter, Abstract, [0010], [0019], [0028], [0050], [0054], [0057], FIGS. 1-2. Annotated Figure 1 below illustrates Felter’s data processing system 100, including processors 102-1 and 102-2 (first and second domains).



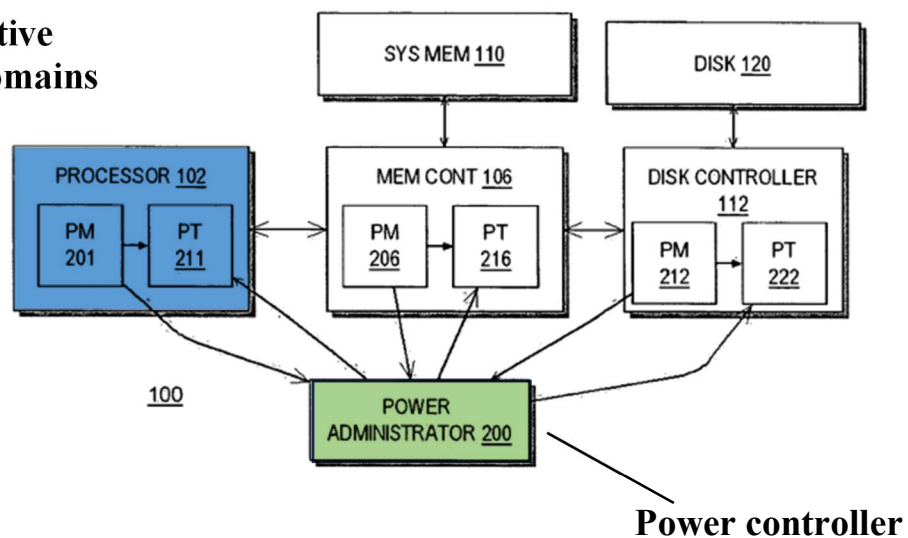
Felter, FIG. 1 (annotated)

75. In my opinion, a POSITA would have understood that Felter's processors 102-1 and 102-2 operate as first and second domains because they are independently operable using distinct operating voltages and frequencies. *See* Felter, [0010] (“[C]onstraining processor activity may be achieved by reducing the voltage and frequency applied to the processor.”), [0034] (“[P]rocessors 102 may support voltage and frequency scaling as part of the processor’s power conservation implementation.”), [0042]-[0049] (describing how to allocate “variable power” to “two independent frequency-scalable processors”), claim 15 (“[T]he instructions for constraining system activity includes instructions for reducing a voltage and frequency applied to the processor.”). This is consistent with the ’316 patent, which uses the term domain “to mean a collection of hardware and/or logic that operates at the same voltage and frequency point.” ’316 patent, 1:52-56.

76. Felter further discloses that its “dynamic power allocation administrator 200 (also referred to herein simply as power administrator 200)” determines “the total power budget (P_{BUDGET}) of the system.” Felter, [0057]. The administrator allocates the available power,” meaning that it budgets power for a current time interval “to system components based on predicted levels of activity for each component.” Felter, [0030], *see also* Felter, [0019] (“The power estimate for each component is used to then allocate the budgeted power amongst all the components.”), FIGS. 2, 5. “When the current interval expires, power administrator

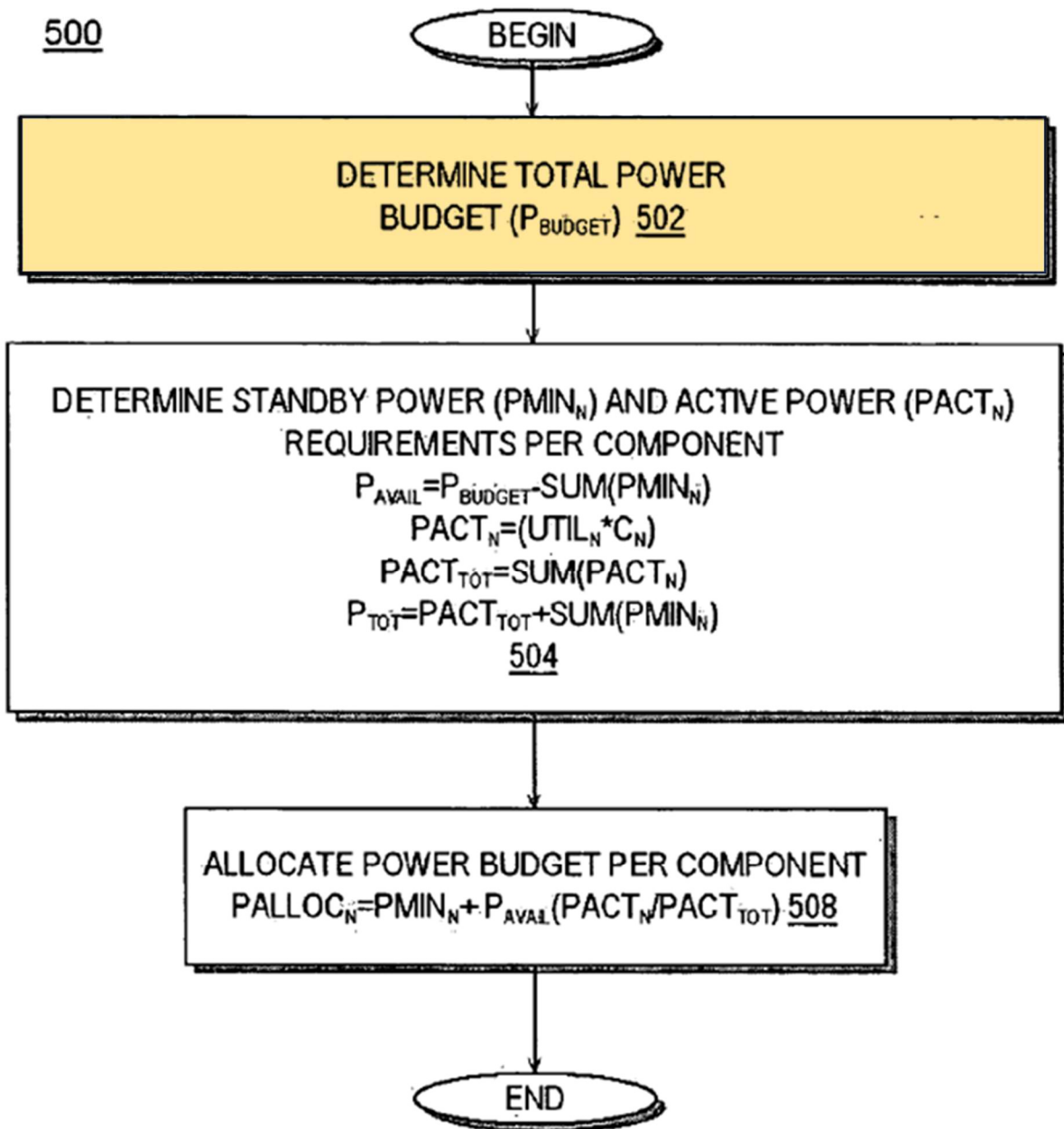
200 generates new predictions for the next upcoming period in block 304 and the process repeats itself indefinitely.” Felter, [0066]. Annotated Figure 2 below illustrates the “power administrator” (power controller) for determining the total power budget (P_{BUDGET}) and the power budget of the first and second domains. See Felter, [0029] (referring to processor 102 as “the processors 102 of Figure 1”).

**Representative
first/second domains**



Felter, FIG. 2 (annotated)

77. Felter’s Figure 5, which I have reproduced below with yellow highlighting, illustrates an exemplary process for distributing power for a current time interval. As block 502 indicates, the process begins by determining the power budget (P_{BUDGET}) for the system. See Felter, [0057] (“In the depicted embodiment, method 500 includes determining (block 502) the total power budget (P_{BUDGET}) of the system.”), FIG. 5.

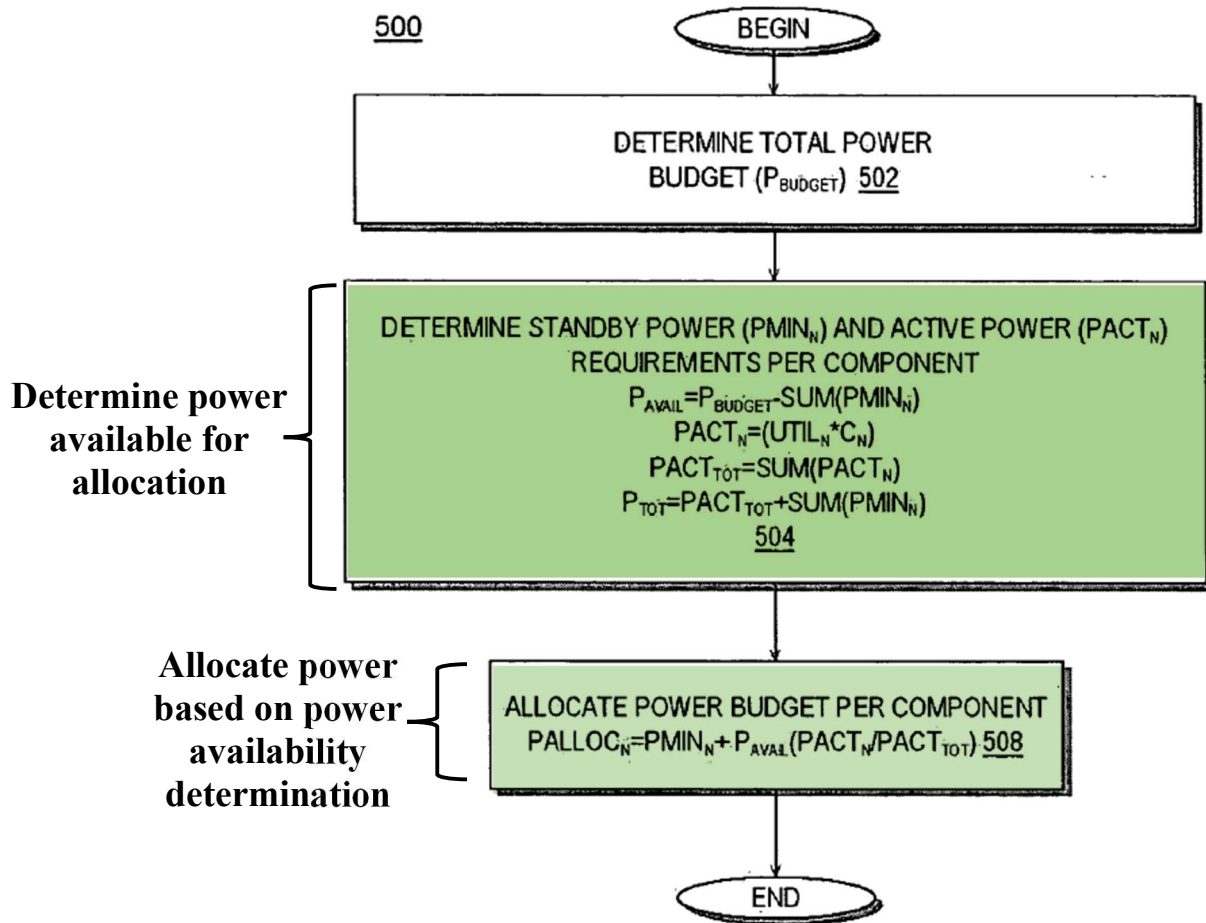


Felter, FIG. 5 (annotated)

c. [8b]: “determining, in the power controller, a portion of the power budget to be allocated to the first and second domains, including”

78. Felter discloses feature [8b] because the power administrator (power controller) uses “the power estimate for each component” to “allocate the budgeted power amongst all the components,” including the “processor cores” (first and second domains). Felter, [0019], [0028]; *see also* Felter, [0026] (“[S]ystem 100 includes first and second general purpose microprocessors 102-1 and 102-2 respectively (generically or collectively referred to herein as processor(s) 102).”), [0037]-[0049] (describing how to allocate power by “two independent frequency-scalable processors using a single memory (DRAM) system” by “DVFS”), [0057]-[0061] (describing how to allocate the budgeted power among N components (domains)), FIG. 5.

79. The budgeted power may be “allocated to the various components” (first and second domains) using Felter’s “modified pro rata technique.” Felter, [0061]. Felter’s Figure 5, which I have reproduced below with highlights, shows that “the amount of power [to be] budgeted for each component P_{ALLOC_N} is equal to the component’s standby power requirements (which must always be met) and a portion of the available power P_{AVAIL} .” Felter, [0061], FIG. 5.



Felter, FIG. 5 (annotated)

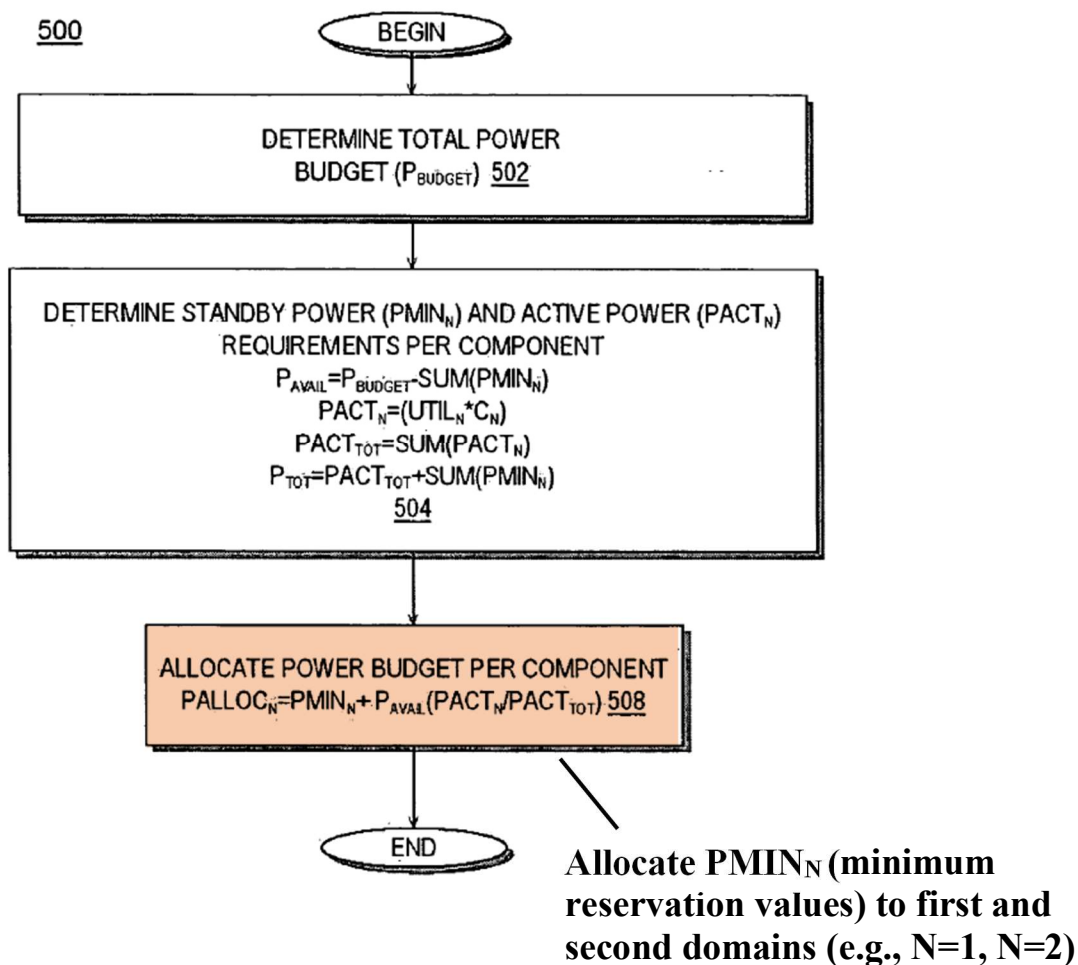
80. As Felter explains, the allocation determination “effectively distributes the available system power to the system components in a manner that (1) ensures that each component receives a minimum level of power and (2) allocates the available active power in proportion to the requirement for the components by the workload(s) executing on the system.” Felter, [0061].

d. [8c]: “allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain, and”

81. Felter discloses feature [8c] because the power administrator allocates “standby power requirements (which must always be met)” to each component (a minimum reservation value). Felter, [0061]; *see also* Felter, [0039] (explaining that “ $P_{proc}=G(f)+P_{min}$ where $G(f)$ is a polynomial in f [requency] . . . , [and] P_{min} is the minimum operational power of the processor at its lowest operating point”), [0042]-[0049] (showing that each power allocation for “two independent frequency-scalable processors” includes the respective P_{min} value of each), [0058] (referring to the P_{min} values as “the standby values (minimum power values) of each system component”), [0065] (“reservoir for minimal component activity”), FIG. 5.

82. Felter explains that each system component, including processors 102-1 and 102-2, has its own respective “fixed” $PMIN_N$ (also called “standby power”) corresponding to “the minimum operational power of the processor [N] at its lowest operating point.” Felter, [0039]; *see also* Felter, [0042]-[0049] (showing that each power allocation for “two independent frequency-scalable processors” includes the respective P_{min} value of each), [0058] (referring to the P_{min} values as “the standby values (minimum power values) of each system component”), [0065] (“reservoir for minimal component activity”), FIG. 5. The power administrator operates “[u]nder the assumption that each component always draws its standby or

minimum power.” Felter, [0058]. After determining the “standby demands of each of the components,” the power administrator allocates to each component its respective “standby power requirements (which must always be met) and a portion of the available power P_{AVAIL} ” (i.e., what remains of the power budget P_{BUDGET} after subtracting out all of the $PMIN_N$ values). Felter, [0061], FIG. 5. As Felter’s Figure 5 shows, which I have reproduced with annotations below, each domain N is allocated its respective “minimum operational power [$PMIN_N$]” (minimum reservation value).



Felter, FIG. 5 (annotated)

83. Felter further states that the power administrator may “ensure there is always enough power for each component to obtain some minimal work from it” by including “a reservoir for minimal component activity” in the standby power allocated to each component. Felter, [0065]. Thus, Felter discloses allocating a minimum reservation value ($PMIN_N$) to each of the processor cores (first and second domains, e.g., where $N=1$ and $N=2$).

- e. **[8d]: “sharing a remaining portion of the power budget according to a first sharing policy value for the first domain and a second sharing policy value for the second domain; and”**

84. Felter discloses feature [8d] because the power administrator (power controller) allocates power to the various components (domains) using a “modified pro rata technique” (as I explain above for feature [8c]) whereby “the amount of power budgeted for *each component* is equal to *the component’s* standby power requirements (which must always be met) *and a portion of the available power* P_{AVAIL} .” Felter, [0058]-[0061] (emphasis added). The portion of available power for each of the processor cores (first and second domains) “is determined by the ratio of *the components active power requirements* $[PACT_N]$ to the total active power requirements of the system $[PACT_{TOT}]$.” Felter, [0061] (emphasis added), FIG. 5 (blocks 504 and 508).

85. After determining “the standby values (minimum power values) of each system component,” Felter teaches to subtract those values from the total power budget (P_{BUDGET}), a quantity Felter refers to as the “available power P_{AVAIL} .” Felter [0058], [0061]. Felter explains: “Using the standby values [P_{MIN_N}], an available power figure (P_{AVAIL}) is determined by subtracting from the maximum power deliverable by the system [P_{BUDGET}] the sum of all the standby values.” Felter, [0058]; *see also* Felter, FIG. 5 (explaining that “ $P_{\text{AVAIL}} = P_{\text{BUDGET}} - \text{SUM}(P_{\text{MIN}_N})$ ”). Felter’s so-called “active power” P_{AVAIL} is what feature [8d] recites—a “remaining portion of the power budget” after “allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain” to satisfy feature [8c].

86. Felter’s power administrator (power controller) then allocates “a portion of the available power P_{AVAIL} ” to each component (domain). Felter, [0061]. Each portion “is determined by the ratio of the components active power requirements [P_{ACT_N}] to the total active power requirements of the system [$P_{\text{ACT}_{\text{TOT}}}$].” Felter, [0061]. “This formula is represented in block 508 as $P_{\text{ALLOC}_N} = P_{\text{MIN}_N} + P_{\text{AVAIL}} * (P_{\text{ACT}_N} / P_{\text{ACT}_{\text{TOT}}})$.” Felter, [0061], FIG. 5 (block 508). A POSITA would have understood that the ratios $P_{\text{ACT}_N} / P_{\text{ACT}_{\text{TOT}}}$ for each domain N correspond to the respective “sharing policy value[s]” recited in feature [8d].

87. Using this “modified pro rata technique,” the power administrator “allocates the available active power in proportion to the requirement for the components by the workload(s) executing on the system.” Felter, [0061].

88. Based on the equation Felter provides in block 508 of FIG. 5, the total power allocated for a first domain (N=1) may be expressed as follows:

$$PALLOC_1 = P_{MIN_1} + P_{AVAIL} * (PACT_1 / PACT_{TOT});$$

where $(PACT_1 / PACT_{TOT})$ is the first sharing policy value for the first domain. Similarly, the total power allocated for a second domain (N=2) may be expressed as follows:

$$PALLOC_2 = P_{MIN_2} + P_{AVAIL} * (PACT_2 / PACT_{TOT});$$

where $(PACT_2 / PACT_{TOT})$ is the second sharing policy value for the second domain. The value of $PACT_N$ for a domain is computed as $UTIL_N * C_N$, which is a sharing policy that shares power in proportion to the utilization of that domain. Felter, [0058], FIG. 5 (block 504).

f. [8e]: “controlling a frequency of the first domain and a frequency of the second domain based on the allocated portions.”

89. Felter discloses feature [8e] because it discloses regulating system component (domain) activity using “dynamic voltage-frequency scaling,” better known by its acronym “DVFS.” *See* Felter, [0036]-[0049], [0065]. After determining a power budget and allocating it among the various components

(domains), Felter's power administrator "enforces the allocated power budget by monitoring the activity of the various components and constraining or throttling the activity of any component that exceeds its power allocation." Felter, [0030]. The power administrator "monitors component activity using performance monitors (PM's)" and "throttles" the processors' activity using "performance throttles (PT)" to reduce each component's "power consumption." Felter, [0031], [0033], [0057], FIGS. 2, 5. When Felter's first or second processor 102-1 or 102-2 "is subject to an activity limit [and] reaches that limit, further activity is prohibited until the interval is over." Felter, [0033]. Felter further explains that one technique for achieving the aforementioned "throttling" is "dynamic voltage-frequency scaling" (also called "DVFS" or "voltage-frequency throttling"), which was a well-known way in which to control the frequency and voltage of a system component. Felter, [0035]-[0036], [0065]; *see also* Felter, [0037]-[0049] (describing a process for using "DVFS" for "two independent frequency-scalable processors using a single memory (DRAM) system"). A POSITA would have readily understood how to perform DVFS, as it was (and still is) the subject of the well-known Advanced Configuration and Platform Interface ("ACPI") standard.

90. Felter discloses in one example that "[t]he actual frequency setting for processor 1 for a power limit can be computed as $G^{-1}(P1PWrlimit - Pmin1)$," where

G is a polynomial function that sets the frequency f from any “known voltage-frequency relationship.” Felter, [0039], [0049].

91. It is therefore my opinion that Felter discloses controlling a frequency of the first domain and a frequency of the second domain based on the allocated portions of the power budget.

2. **Claim 12: “ The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”**

92. Felter discloses, or at least renders obvious, the additional features of claim 12 because it discloses “throttling” (not granting) a processor core’s activity when it attempts to perform activity that would “exceed” its budget-related “limit” during a particular time interval. Felter, [0065]. In response to such a request for higher frequency being denied (by denying the processor core the ability to perform more instructions during the allotted time period), a POSITA would have at least found it obvious based on Felter’s disclosures to increase that processor core’s frequency during a subsequent time interval by allocating more power to it. *See* Felter, [0051] (“In perhaps the simplest implementation, the predicted activity for time period N+1 is the measured activity for time period N. This approach relies on the not unreasonable assumption that very recent activity is a good predictor of

current activity. This approach is also desirable for its responsiveness in the context of a rapidly changing operating environment.”), [0066] (“When the current interval expires, power administrator 200 generates new predictions for the next upcoming period in block 304 and the process repeats itself indefinitely.”).

93. As I explain above for feature [8c], Felter discloses a “modified pro rata technique” for allocating power to each component (domain). Felter, [0061]. According to this technique, Felter’s power administrator (power controller) allocates power in a way that accounts for each “component’s standby power requirements (which must always be met) and a portion of the available power P_{AVAIL} .” Felter, [0061]. This “represents a technique for distributing scarce system power equitably so that the system performance of each component is impacted proportionately” and “in proportion to the requirement for the components by the workload(s) executing on the system.” Felter, [0061].

94. Felter further discloses that if either of the components “exceed” their corresponding limit during a given interval—that is, it exceeds its allocated power budget—then the power administrator throttles that component’s activity “until the next monitoring interval begins.” Felter, [0065]. “When the current interval expires, power administrator 200 generates new predictions for the next upcoming period” and allocates power to the throttled components according to their new needs. Felter, [0066], *see also* Felter [0051]-[0054] (“[P]ower

administrator 200 includes allocating (block 306) power budgets to the various system components based on the predicted level of activity.”), [0059] (“Specifically, the active power predicted for a component N is indicated as the product of the activity (UTIL) predicted for component N ($UTIL_N$) and a constant of proportionality for component N (C_N).”), FIG. 3 (block 306), FIG. 5 (block 504).

95. To that end, Felter explains that “the predicted activity for time period N+1 is a simple function of the monitored activity for time period N.” Felter, [0051].

In one example, Felter explains the following:

[T]he predicted activity for time period N+1 is the measured activity for time period N. This approach relies on the not unreasonable assumption that very recent activity is a good predictor of current activity. This approach is also desirable for its responsiveness in the context of a rapidly changing operating environment.

Felter, [0051]. Since Felter teaches to use the amount of recent activity to predict utilization for the current interval, a POSITA would have understood that if one of Felter’s components (domains) was throttled during the previous time interval, then its predicted utilization ($UTIL_N$) should be incremented for the next interval. *See* Felter, [0051] (“[T]he predicted activity for time period N+1 is a simple function of the monitored activity for time period N (or for another recent time period).”), [0054] (“[P]ower administrator 200 includes allocating (block 306) power budgets to the various system components based on the predicted level of activity.”), [0066]

(“When the current interval expires, power administrator 200 generates new predictions for the next upcoming period in block 304 and the process repeats itself indefinitely.”), FIG. 3. In other words, if one of Felter’s components (domains) was prevented from performing more tasks during the previous time interval (denying its request to perform at a higher frequency), then a POSITA would have understood from Felter’s disclosures that the component (domain) should be granted a higher frequency during the next budgeted time interval.

96. It was well known in the art to increase the frequency of a computing domain when its power needs are not satisfied. For example, Therien describes a “policy manager” that “periodically evaluates” a system “to determine whether any power budget adjustments are appropriate” and “make[s] budget adjustments as conditions associated with the system 400 change (e.g., when a component raises or lowers an increase request flag).” Therien, [0031]. Similarly, Conroy explains that “the power of the system may be shifted toward the subsystem having a higher workload” such that “higher power draw is allocated to and allowed for the subsystem having a higher workload.” Conroy, [0006]. This is also common sense. A component that is getting busier than it was before should be given additional resources to work faster if it is unable to keep pace with system demands.

97. Accordingly, Felter discloses, or at least renders obvious, that its first sharing policy value ($PACT_1/P_{TOT}$, which is proportional to $UTIL_1$) should be

incremented when a request for a higher frequency for the first domain is not granted (throttled), and the second sharing policy value ($PACT_2/P_{TOT}$, which is proportional to $UTIL_2$) is to be incremented when a request for a higher frequency for the second domain is not granted (throttled).

B. Ground 2: Claims 13, 14, and 17 Would Have Been Obvious Based on Finkelstein

1. Independent Claim 13

a. [13pre]: “a system comprising:”

98. Finkelstein discloses feature [13pre] because it discloses a “computing system” with, among other things, “one or more processors,” a “power source,” “voltage regulator(s),” and “power management logic.” Finkelstein, [0010]-[0014]; *see also* Finkelstein, [0042]-[0047], FIGS. 1, 6. Finkelstein’s Figure 1, reproduced below, illustrates an exemplary computing system 100.

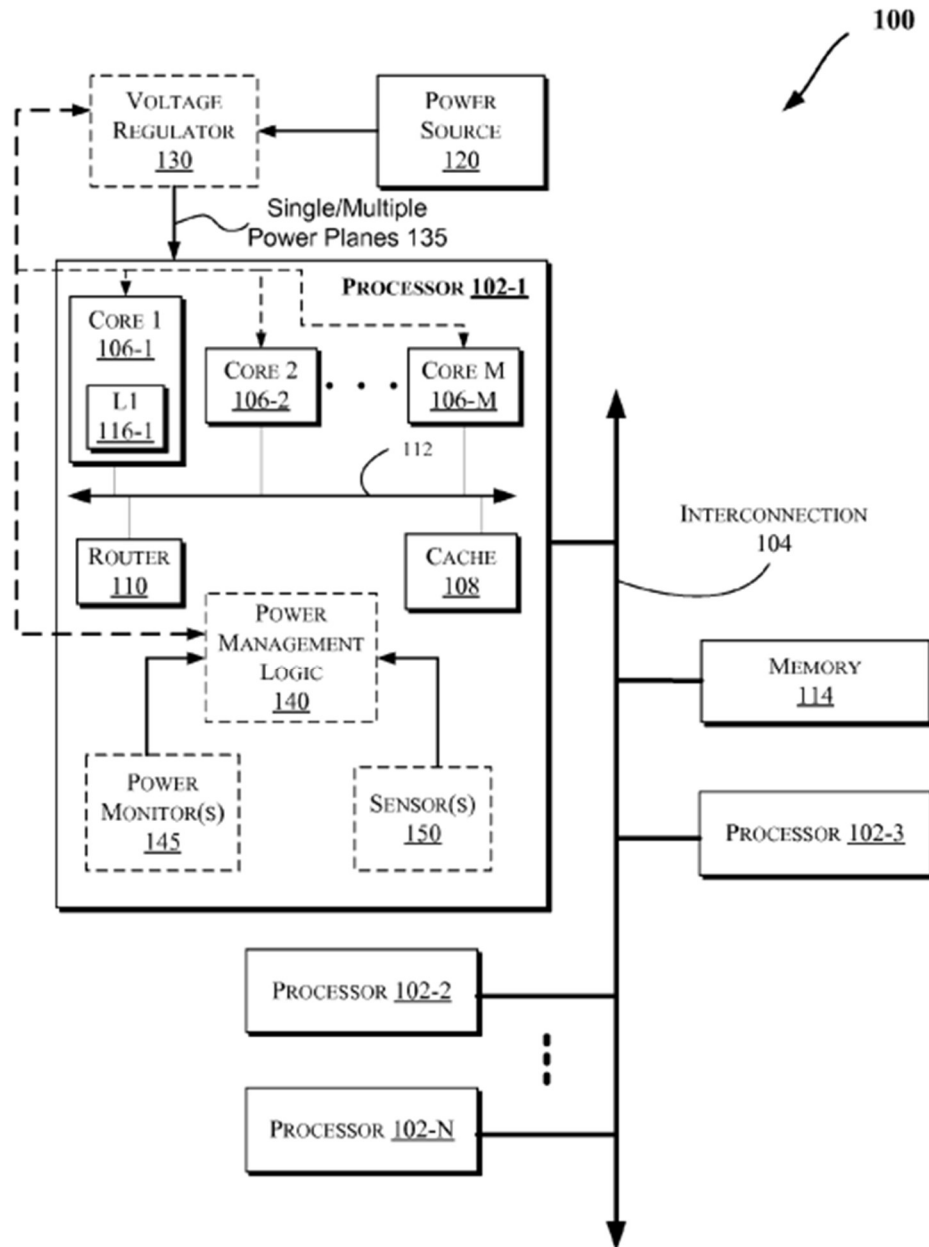


FIG. 1

Finkelstein, FIG. 1

b. [13a]: “a multicore processor having a first domain including a plurality of cores, a second domain including a graphics engine, and a third domain including system agent circuitry,”

99. Finklestein discloses, or at least renders obvious, feature [13a] because its exemplary computing system comprises a multicore, “multiple power plane” processor 102-1. Finkelstein’s processor 102-1 includes, among other things, the following power domains: (1) a first power plane (domain) to power at least first and second processor cores 106-1 and 106-2, (2) a second power plane (domain) to power a “graphics . . . controller[]” (graphics engine) that facilitates “Graphics Effect(s) (GFX),” and (3) a third power plane (domain) to power Finkelstein’s “power management logic” (system agent circuitry). Finklestein, [0011], [0014], [0026], [0046], FIGS. 1, 6.

100. Finkelstein discloses a first domain including a first power plane to power a plurality of processor cores. Finkelstein’s processor 102-1 may include “multiple power planes 135 (e.g., where each power plane may supply power to a different core or group of cores).” Finkelstein, [0014]; *see also* Finkelstein, [0009] (“[I]n designs with multiple power planes, power may be distributed among different power planes”), [0011] (“[T]he processor 102-1 may include one or more processor cores 106-1 through 106-M”), [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of for example, unused

processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”), FIG. 1. Each power plane is prescribed its own “power/energy budget” so that it “may supply power to a different core or group of cores.” Finkelstein, [0014], [0026]. The ’316 patent uses the term domain “to mean a collection of hardware and/or logic that operates at the same voltage and frequency point.” ’316 patent, 1:52-56. A POSITA thus would have understood that Finkelstein’s power plane and respective processor cores operate on their own domain. *See also* Finkelstein, claim 6 (reciting “a voltage regulator to supply power to more than one cores of the processor via the single power plane”). In Finkelstein, one power plane may supply “processor core power” to a group of cores (a first domain), and another power plane may supply power to a “graphics . . . controller[]” to facilitate “Graphics Effect(s) (GFX) performance,” as I further explain below. Finkelstein, [0011], [0026].

101. Finkelstein discloses a second domain including a second power plane to power a graphics controller (graphics engine). Finkelstein further discloses that processor 102-1 may include a power plane that includes a “graphics . . . controller[]” comprising a “graphics and memory control hub (GMCH)” with “a graphics interface 614 that communicates with a graphics accelerator 616” over “an accelerated graphics port (AGP)” (collectively, a second domain including a graphics engine). Finkelstein, [0011], [0044]; *see also* Finkelstein, [0046]

(explaining that “the processor[] and the GMCH[] may be combined to form a single chip” and that “the graphics accelerator 616 may be included within the GMCH 608 in other embodiments of the invention”), FIG. 6. The first power domain for the processor cores (first domain) and the second power plane for the graphics controller (second domain) may “share a common package power/energy budget” such that “unused processor core power” may be used “for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.” Finkelstein, [0026]. Although the ’316 patent does not provide an express definition of “graphics engine,” it does make clear that a graphics engine performs “a graphics intensive workload,” as opposed to “a computing intensive workload.” ’316 patent, 6:23-26. Finkelstein’s second power plane for the graphics controller (second domain) includes the features highlighted in annotated Figure 6 below. *See* Finkelstein, [0011] (“[T]he chip may include one or more . . . graphics and/or memory controllers (such as those discussed with reference to FIGS. 6-7) . . .”). [0044] (“The GMCH 608 may also include a graphics interface 614 that communicates with a graphics accelerator 616 . . . via an accelerated graphics port (AGP).”), [0046] (“[T]he processor 602 and the GMCH 608 may be combined to form a single chip. Furthermore, the graphics accelerator 616 may be included within the GMCH 608 . . .”).

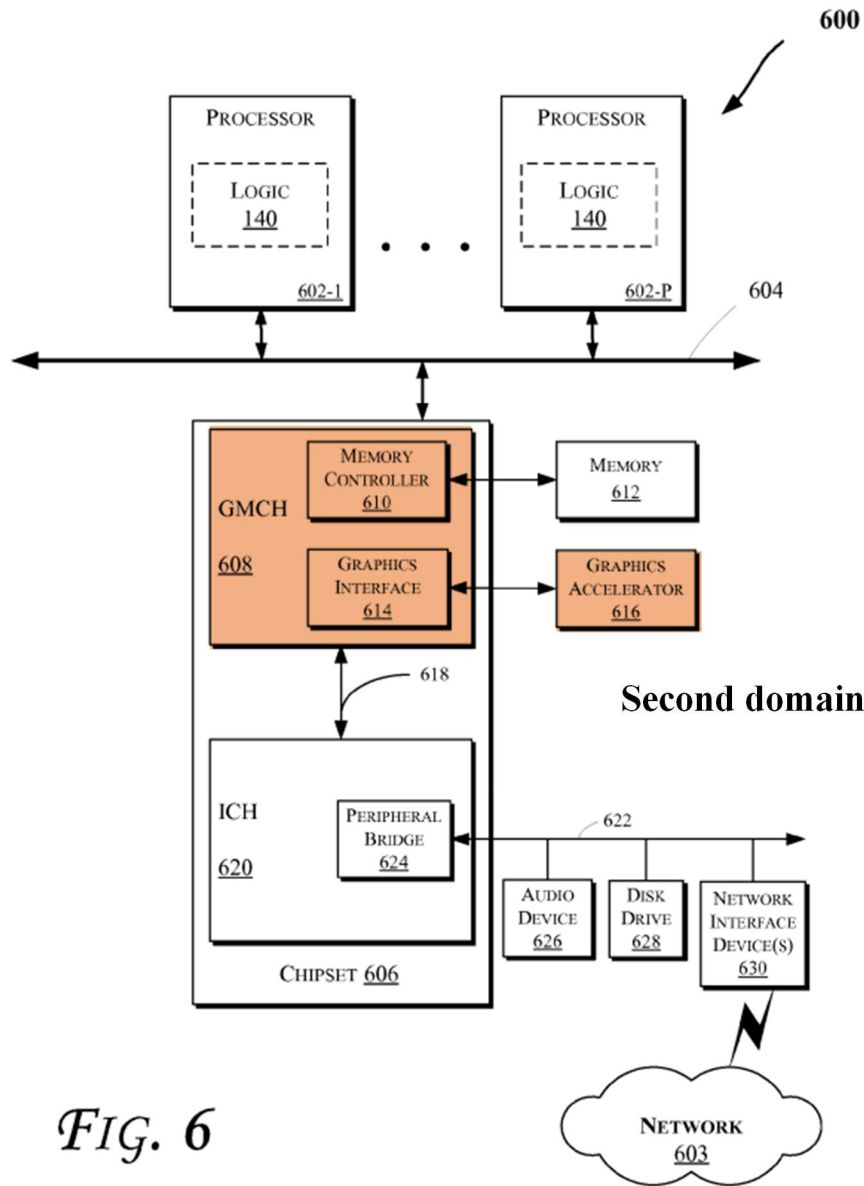


FIG. 6

Finkelstein, FIG. 6 (annotated)

102. Finkelstein discloses a third domain including power management logic (system agent circuitry). Finklestein's third domain, like the "system agent domain" of the '316 patent, includes "power management logic" (system agent circuitry) that allows it to manage power budgeting and allocation for multiple power planes using

power monitors, sensors, and voltage regulators. *Compare* '316 patent, 9:10-15 (explaining “system agent domain 450 may execute at a fixed frequency and may remain powered on at all times to handle power control events and power management”), 3:16-62 (“Embodiments may dynamically calculate a package power budget which is a metric that measures the power headroom available to the processor for a given time interval. . . . [E]nergy can be obtained from reading external current and voltage monitoring sensors such as current monitoring circuitry implemented in a voltage regulator”), *with* Finkelstein, [0039] (explaining power management logic 140 “determine[s] future values for each power plane budget during a next time period”), [0016] (“[T]he logic 140 may be coupled to receive information . . . to indicate status of one or more sensors 150 (where the sensor(s) 150 may be provided proximate to components of system 100 . . . to sense variations in temperature, operating frequency, operating voltage, power consumption, inter-core communication activity, etc.) and/or information from one or more power monitoring logics 145 (e.g., which may indicate the operational status of various components of system 100 such as operating temperature, operating frequency, operating voltage, operating status (e.g., active or inactive), power consumption (instantly or over a period of time), etc.).”). A POSITA would have understood performance of this task must be guaranteed because it is essential to proper system operation.

103. Finklestein's power management logic manages the energy budget to be consumed by each power plane and its cores (domains). *See* Finkelstein, [0039] (explaining that power management logic 140 determines the energy budgets at operation 502 (formula (1)), allocates the energy budgets to the relevant power planes at operation 504 (formula (2)), and determines the power states of each component at operation 506 (formula (3))), FIG. 5. A POSITA would have understood that, like all components of processor 102-1, power management logic 140 consumes power in order to function. *See* Finkelstein, [0014] ("The system 100 may also include a power source 120 . . ."). A POSITA also would have understood that Finkelstein's power management logic 140 operates on a fixed, independent power budget (as I explain further for feature [13b]) so that it can reliably execute its power management functions. *See* Finkelstein, [0016]-[0017] & [0026] (describing how power management logic 140 manages power for system components like processor cores and graphics accelerators with variable tasks to perform but not for components like power management logic 140 itself with fixed tasks to perform), [0039] (describing the power management functions performed by power management logic 140). Thus, a POSITA would have at least found it obvious that Finkelstein's power management logic operates on a third domain within the meaning of the '316 patent, as I further describe for feature [13b] below.

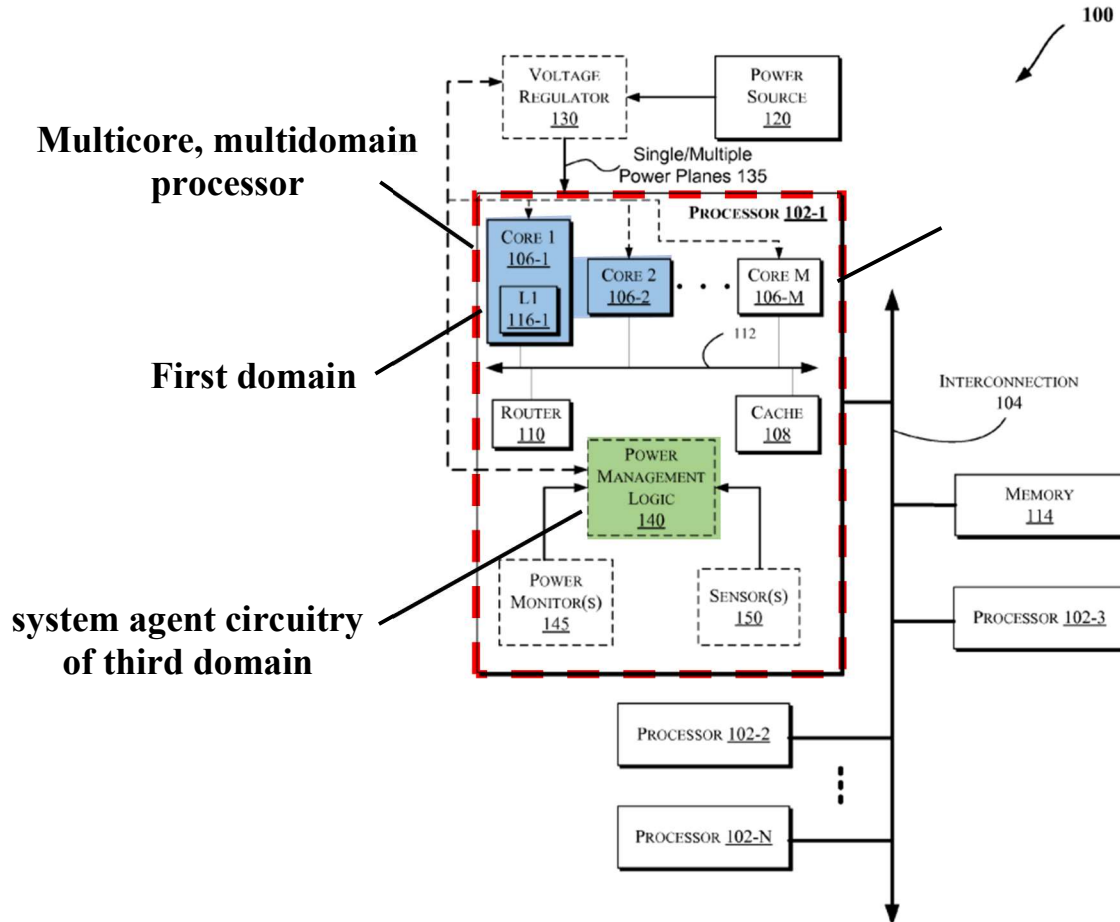


FIG. 1

Finkelstein, FIG. 1 (annotated)

104. Finkelstein's Figure 1, which I have annotated above, illustrates a multicore, "multiple plane" processor 102-1 that includes a first power plane for a group of cores 106-1 and 106-2¹ (a first domain, *see* Finkelstein, [0014]), a second

¹ My designation of cores 106-1 and 106-2 as the claimed "group of cores" operating on the same power plane is illustrative only. Any plurality of grouped cores 106-m

power plane for a “graphics . . . controller[]” that performs “Graphics Effect(s) (GFX)” (a second domain, *see* Finkelstein, [0026]), and a third power plane for power management logic (a third domain, *see* Finkelstein, FIG. 1). The power management logic (system agent circuitry) includes a power sharing logic to determine the variable power budget for a current time interval, as I explain below for feature [13c].

105. Although not expressly illustrated in Finkelstein’s Figure 1, Finkelstein explains that “the chip [in Figure 1] may include one or more . . . graphics and/or memory controllers (such as those discussed with reference to FIGS. 6-7)” Finkelstein, [0011]. Finkelstein’s Figure 6, in turn, which I have annotated below, illustrates a “graphics and memory control hub (GMCH) 608” that includes “a graphics interface 614 that communicates with a graphics accelerator 616 . . . via an accelerated graphics port (AGP).” Finkelstein, [0043]-[0044]. According to Finkelstein, “the processor . . . and the GMCH 608 may be combined to form a single chip. Furthermore, the graphics accelerator 616 may be included within the GMCH 608.” Finkelstein, [0046]; *see also* Lalonde, [0022]-[0027], FIG. 5 (collectively describing the same architecture as Finkelstein’s FIG. 6 with an integrated graphics

through 106-n would correspond to the claimed “plurality of cores” associated with the “first domain.”

controller). Because “it is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload,” a POSITA would have understood an on-chip graphics power plane is distinct from the processor power plane. Finkelstein, [0026]. And because the graphics and processor workloads are variable, unlike the tasks assigned to the power management logic, a POSITA would have understood the processor and graphics power planes are distinct from the power plane for the power management logic.

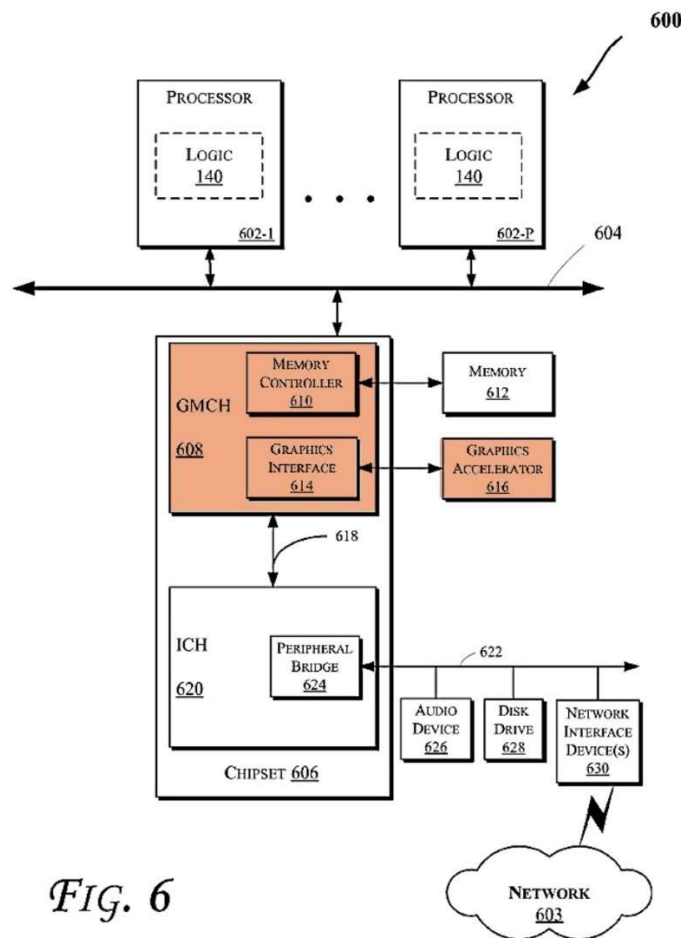


FIG. 6

Finkelstein, FIG. 6 (annotated)

c. [13b]: “the third domain to operate at a fixed power budget and to dynamically allocate a variable power budget between the first and second domains, wherein”

106. Finklestein discloses, or at least renders obvious, feature [13b] because the third domain (power plane for the power management logic) dynamically allocates a variable power budget (E^k) between the first domain (processor core power plane) and second domain (graphics controller power plane) using a pro rata technique. *See* Finkelstein, [0026]-[0029], [0034], [0039]. Additionally, a POSITA would have understood it operates on a fixed power budget, as was routine in the art for such power management logic.

107. First, Finkelstein discloses the third domain dynamically allocating a variable power budget between the first and second domain.

108. For example, Finkelstein’s power management logic receives information about various constraints affecting energy consumption and uses those constraints (denoted by the subscript or superscript “k”) to determine a power budget for each constraint, which may be “a common package power/energy budget” shared by the processor core and graphics controller power planes. Finkelstein, [0026]; *see also* Finkelstein, [0039] (“[A]t an operation 502, the energy budget remainder per constraint E_n^k may be determined (e.g., by the logic 140 in accordance with formula (1) . . .).”). Finkelstein further explains that “[f]or constraint k, user (or application

defined) preferences” denoted by “vectors W_k ” are used to “describe how budget E^k is distributed among” the associated power planes i that are subject to constraint k .² Finkelstein, [0034]. The portion of the power budget (E^k) that is allocated to power plane i is denoted $E^{k,i}$, and it is assigned on a pro rata basis using the following equation: “ $E^{k,i}=W_{ki}E^k$.” Finkelstein, [0034]; *see also* Finkelstein, [0039] (“At an operation 504, the energy budget remainder per power plane $E_n^{k,i}$ may be determined (e.g., by the logic 140 in accordance with formula (2) above).”).

² Although Finkelstein allocates an energy budget, a POSITA would have understood it is a “power budget” according to the ’316 patent. I discuss this further in Section VIII.A above. For example, a POSITA would have understood a power budget can be converted to an energy budget, and vice versa, simply by multiplying the power budget by the time interval (or dividing the energy budget by the time interval). *See* Finkelstein, [0028] (“ $E_{n+1}=\alpha E_n+(TDP_n-P_n)\Delta t_n$ ”). The ’316 patent itself also makes clear that the claimed “power budget” may be expressed in terms of energy (using Joules), not just in terms of power (Watts). *See* ’316 patent, 3:22-62 (describing equation [1]), claim 7 (defining “the power budget for the current time interval” as an energy “ E_n ” and saying “to determine the power budget” in terms of “energy”). A POSITA thus would have understood the claimed “power budget” refers interchangeably to power (in Watts) and energy (in Joules).

109. Accordingly, Finkelstein discloses that its power management logic dynamically allocates the power budget between the first and second power domains (the processor core and graphics controller power planes, respectively).

110. Second, a POSITA would have understood that Finkelstein's power management logic operates on a fixed power budget so that it is able "to control supply of power to components of the processor." Finkelstein, [0016]. To accomplish this, a POSITA would have understood the power management logic requires a steady stream of power (a fixed power budget)—e.g., to ensure it is able to sense "operating frequency, operating voltage, power consumption, [and] inter-core communication activity" and "instruct[ing] the [voltage regulator(s) 130, power source 120, and/or individual components of system 100 (such as the cores 106) to modify their operations." Finkelstein, [0016], [0024], [0031]. Otherwise, proper system performance would be compromised.

111. Finkelstein also discloses that its power management logic performs the same function in every time step. *See* Finkelstein, [0039] ("At operation 508, the time step may be incremented (e.g., by logic 140), e.g., to determine future values for each power plane budget during a next time period."), FIG. 5. To a POSITA, this would have indicated the power management logic runs on the same power budget each time interval, and thus would have expected Finkelstein's power management logic to operate using a fixed power budget.

112. Additionally, power management logic 140 in Finkelstein's Figure 1 controls the output of the voltage regulators but it does not change its own power. The voltage regulator controls the processor cores, for example, but does not change its own power setting. In other words, a POSITA would have understood that power management logic 140 operates on a fixed power budget.

113. I have been asked to consider whether a POSITA would have found it obvious to configure Finkelstein's logic to operate on a fixed power. In my opinion, a POSITA would have found it obvious. For example, a POSITA would have been motivated to configure Finkelstein's power management logic to operate on a fixed power budget to ensure proper system performance, including by ensuring the power management logic is capable of consistently and reliably managing power to the other components. It was well known and routine in the art to operate such power management logic using a fixed power budget. A POSITA would have understood it is a "non-throttled component" that uses a fixed power budget, as opposed to a "throttled" component, such as a processor core or graphics controller, that has different performance levels. *See, e.g.,* Conroy, [0081] ("[T]he system includes throttled component(s) and non-throttled component(s). A throttled component has different throttle settings at which the component is functional but at different power/performance levels (operating setting). . . ."), [0203] (describing how to allocate power for computation-heavy and graphics-heavy workloads). A POSITA

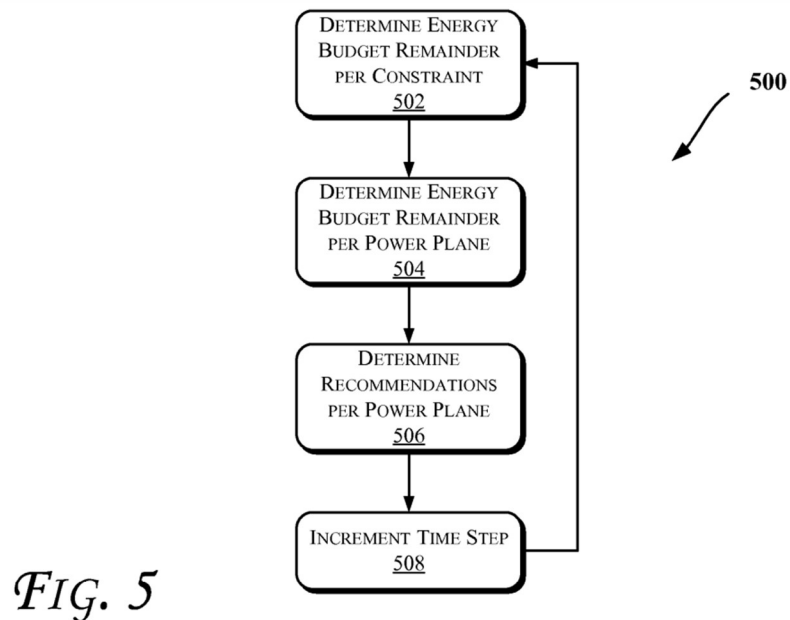
would have understood, for example, to implement Finkelstein's power management logic with an integrated "microcontroller" that "is used to budget the power usage dynamically . . . to the throttled component(s)" and "can be considered as part of the non-throttled components." Conroy, [0094]; *see also* Felter, [0030] (explaining that Felter's "power administrator 200," which is analogous to Finkelstein's power management logic 140, "may be implemented as an integrated or separate on-chip/on-board programmable or hard-coded microcontroller").

114. In my opinion, a POSITA also would have had a reasonable expectation of success when running Finkelstein's power management logic on a fixed power budget because such a configuration would have only required routine skill in the art to implement. For example, a POSITA would have known that "a simple worst-case analysis (adding together the maximum values which could happen under any conditions)," conventional look-up tables, or other "analysis based on the information provided by the subsystems and detailed knowledge of the subsystem's state" could be used to allocate a fixed power budget to Finkelstein's power management logic. Conroy, [0146]; *see also* Conroy, [0165] ("FIG. 11 illustrates a table (1101) to look up the power usage requirement of the non-throttled component"). Again, this was well known and routine in the art. *See, e.g.*, Conroy, [0081] ("[T]he system includes throttled component(s) and non-throttled component(s). A throttled component has different throttle settings at which the component is

functional but at different power/performance levels (operating setting). . . . ”), [0094] (“[A] *microcontroller (411) is used to budget the power usage dynamically.* The power supply (401) (e.g., battery, AC adapter, etc.) provides power to the throttled component(s) (409) (e.g., CPU) and the non-throttled component(s) (405) (e.g., hard drive, DVD ROM, etc.). *The microcontroller (411) can be considered as part of the non-throttled components.* Alternatively, the microcontroller (411) may draw [fixed] power from a power supply different from the power supply (401).” (emphasis added)).

d. [13c]: “the system agent circuitry includes a power sharing logic to determine the variable power budget for a current time interval and”

115. Finkelstein discloses feature [13c] because its power management logic (power sharing logic) “define[s]” and “manage[s]” a variable power/energy budget that is “defined iteratively” from one interval to the next. Finkelstein, [0028]-[0029]; *see also* Finkelstein, [0016]. For example, Finkelstein explains that “the energy budget remainder per constraint E_n^k may be determined (e.g., by the logic 140 in accordance with formula (1) . . .).” Finkelstein, [0039]; *see also* Finkelstein, [0028]-[0029] (providing formula (1): “ $E_{n+1} = \alpha E_n + (TDP_n - P_n) \Delta t_n$ ”), FIG. 5.



Finkelstein, FIG. 5

116. Accordingly, Finkelstein discloses a power sharing logic (e.g., power management logic 140) for determining the variable power budget (E_{n+1}^k) for a current time interval (indicated by the index “n+1”). *See* Finkelstein, [0028]-[0029].

- e. [13d]: “to allocate a first portion of the variable power budget to the first domain according to a first power sharing value for the first domain, and to allocate a second portion of the variable power budget to the second domain according to a second power sharing value for the second domain; and”

117. Finkelstein discloses feature [13d] because its power management logic is configured to distribute first and second portions of the budget E^k —each denoted as $E^{k,i}$ ($i=1$ and $i=2$, for example)—to the processor core power plane and the graphics controller power plane (first and second domains, respectively). The power management logic does this by multiplying the budget E^k by the vector preferences

designated W_{ki} , which correspond to the power sharing values for the first and second domains (e.g., for $i=1$ and $i=2$, respectively).

118. As I explain above for feature [13b], Finkelstein's power management logic (system agent circuitry) multiplies the power budget (E^k) by "user (or application defined) preferences" (W_{ki}) to determine a portion ($E^{k,i}$) of the power budget (E^k) allocated to each power plane i . *See* Finkelstein, [0034] (defining formula (2) as " $E^{k,i}=W_{ki}E^k$ "), [0039] ("At an operation 504, the energy budget remainder per power plane $E_n^{k,i}$ may be determined (e.g., by the logic 140 in accordance with formula (2) above)."). By applying this pro rata power allocation technique to a shared power budget, Finkelstein explains that the power management logic "satisfie[s] . . . both individual constraints per power plane and global constraints per package." Finkelstein, [0026]. For instance, a shared power budget may be allocated between the processor core power plane (first domain) and the graphics controller power plane (second domain). *See* Finkelstein, [0026] ("[I]t is possible to share a common package power/energy budget and allow for use of, for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.").

119. Accordingly, for the first and second power planes (e.g., $i=1$, $i=2$), Finkelstein discloses allocating a first portion ($E^{k,1}$) of the power budget (E^k)

according to a first sharing policy value (W_{k1}) for the first domain and allocating a second portion ($E^{k,2}$) of the power budget (E^k) according to a second sharing policy value (W_{k2}) for the second domain. *See* Finkelstein, [0026], [0034], [0039].

f. [13e]: “a dynamic random access memory (DRAM) coupled to the multicore processor.”

120. Finkelstein discloses feature [13e] because its computing system includes “memory” that “may communicate with the processors . . . via [an] interconnection.” Finkelstein, [0013]. It “may store data, including sequences of instructions that are executed by the processor” and may comprise “dynamic RAM (DRAM).” Finkelstein, [0043].

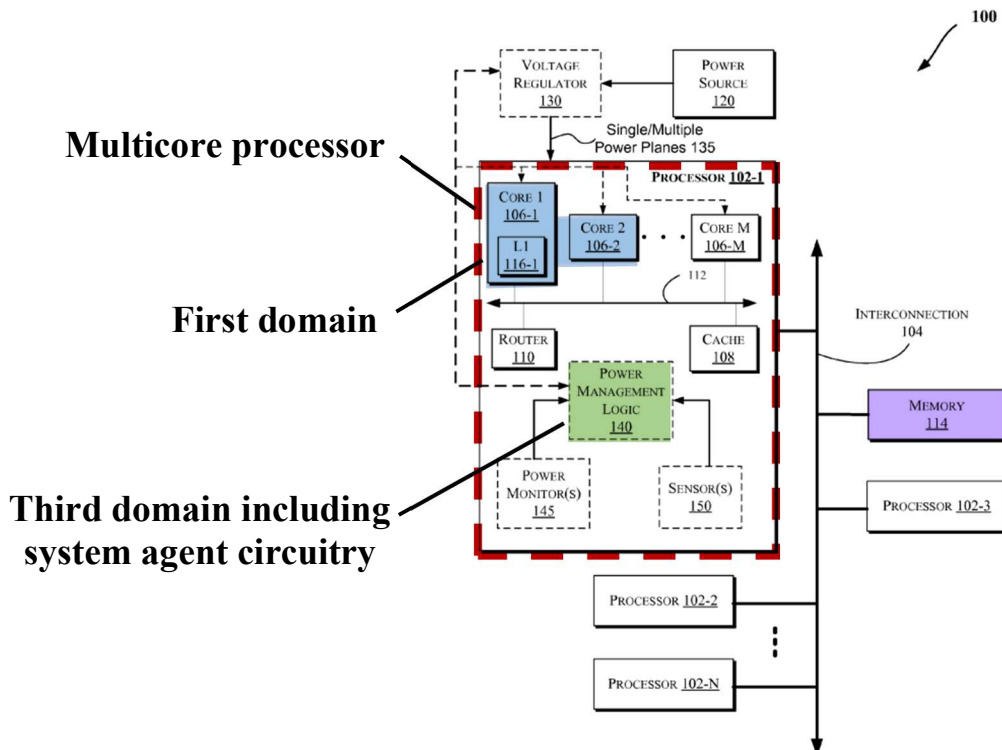


FIG. 1

Finkelstein, FIG. 1 (annotated)

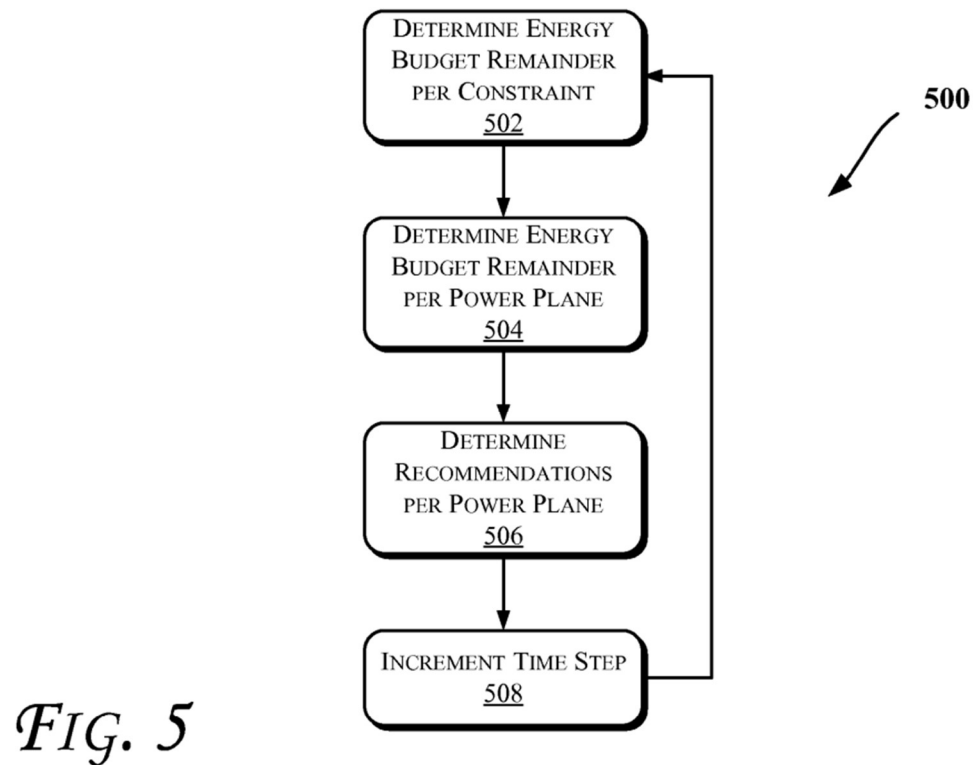
2. **Claim 14: “The system of claim 13, wherein the power sharing logic is to dynamically allocate substantially all of the variable power budget to the first domain for a first workload, and to dynamically allocate substantially all of the variable power budget to the second domain for a second workload executed after the first workload.”**

121. Finkelstein discloses, or at least renders obvious, the additional features of claim 14 because Finkelstein’s power management logic distributes the power/energy budget in a “temporal” way that accounts for the workload during each interval. Finkelstein, [0027]-[0029]. Moreover, the power management logic allocates the “entire budget” to a “single power plane” when needed. Finkelstein, [0034]; *see also* Finkelstein, [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”).

122. **First**, Finkelstein discloses that its power management logic 140 may repeat its power management functions “continuously (or on a periodic basis) . . . for multiple cores sharing the same power plane.” Finkelstein, [0025]; *see also* Finkelstein, [0040], FIGS. 3, 5. For example, Finkelstein explains that “the time step may be incremented (e.g., by logic 140), e.g., to determine future values for each power plane budget during a next time period.” Finkelstein, [0039]; *see also* Finkelstein, FIG. 5. As I explain for feature [13b] above, for example, Finkelstein’s

power management logic does so by dynamically allocating a variable power budget (E^k) between the processor core power plane (first domain) and graphics controller power plane (second domain) using a pro rata technique. *See* Finkelstein, [0028]-[0029] (disclosing formula (1): “ $E_{n+1} = \alpha E_n - (TDP_n - P_n) \Delta t_n$ ”), [0034] (disclosing formula (2): “ $E^{k,i} = W_{ki} E^k$ ”), [0039] (“[A]t an operation 502, the energy budget remainder per constraint E_n^k may be determined (e.g., by the logic 140 in accordance with formula (1) above). At an operation 504, the energy budget remainder per power plane $E_n^{k,i}$ may be determined (e.g., by the logic 140 in accordance with formula (2) above).”). With this technique, “it is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.” Finkelstein, [0026]. Thus, a POSITA would have understood that Finkelstein’s power management logic uses the workload performed during a first time period to dynamically allocate a variable power budget to the first and second domains. Likewise, a POSITA would have understood that Finkelstein’s power management logic subsequently uses the workload performed during a second time period to dynamically re-allocate a variable power budget to the first and second domains. Finkelstein’s Figure 5, which I have reproduced below, illustrates one such method. The power management logic determines a variable power budget “for multiple cores sharing the same power

plane” and allocates a portion of it among those power planes “continuously (or on a periodic basis).” Finkelstein, [0025]; *see also* Finkelstein, [0040], FIGS. 3, 5.



Finkelstein, FIG. 5

123. **Second**, Finkelstein at least suggests that substantially all of the variable power budget would be allocated to the first domain under certain conditions. For example, a first workload during a first time interval may require the processor cores to perform substantially all of the necessary tasks because the graphics controller would be unable or inefficient in doing so. Then, during a subsequent time interval, a second workload may require the graphics controller to perform substantially all of the necessary tasks because the processor cores would

be unable or inefficient in doing so. For example, Finkelstein teaches to share a “common package power/energy budget” between the processing and graphics power planes and explains that “power management may need to be satisfied . . . per package.” Finkelstein, [0026]. According to Finkelstein, “the entire” shared package power budget also may be allocated to one of the power planes during a given time interval. For example, “a *single power plane* may obtain *the entire budget*” based on “user (or application defined) preferences that describe how budget E^k is distributed among power planes.” Finkelstein, [0034]. Finkelstein thus at least suggests that the power management logic would allocate substantially all of the variable power budget to one power plane (e.g., the processor cores) when the “user (or application defined) preferences” (i.e., the W_{ki} values) for the first workload make it appropriate to do so.

124. Finkelstein also explains that “it is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.” Finkelstein [0026]. Based on this disclosure, for example, a POSITA would have understood that the processor cores (first domain) do not require very much power when the system is given a graphics-heavy workload. Substantially all of the power budget would be allocated to the graphics controller (second domain), which only makes sense for a graphics-heavy

workload. Likewise, for a processor-heavy workload, a POSITA would have understood that substantially all of the power budget would be allocated to the processor cores rather than the graphics controller.

125. This type of alternation between computation-heavy and graphics-heavy workloads was well-known in the art, and it is common sense. *See* Gwennap, 6 (“If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs.”); Therien, [0017] (“[A] computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU.”); Bose, Abstract (“During the first mode, instructions associated with applications are executed by a first asymmetric core, and a second asymmetric core is inactive. During the second mode, the instructions are executed by the second asymmetric core, and the first asymmetric core is inactive.”), [0010] (disclosing “reassigning threads of the workload to cores remaining active by assigning threads based on priority constraints and thread execution history to improve the operational efficiency of the multi-core architecture”), [0021] (“[C]ore activation/ deactivation may be provided based upon hardware or priority constraints.”), [0085] (describing a “thread

scheduler” that “priorit[izes]” and “assigns” power distribution to “high performance threads” over “low performance” threads).

126. For example, a POSITA would have understood that many applications, like program development tools and scientific applications, produce a heavy load for the processors and almost no load for the graphics controller. *See e.g.*, Conroy, [0196] (“For example, program development tools and scientific applications present a high load to the CPU, but almost no load to the GPU that leads to an asymmetric load profile of the system (e.g. the CPU consumes a lot more power than the GPU).”). Other applications, such as advanced user interfaces or graphics editing applications, present the opposite situation. Conroy, [0196] (“Advanced user interfaces or graphics editing application present a high load to the GPU and a modest load to the CPU that leads to another asymmetric load profile to the system.”). Accordingly, “[m]any professional applications present an alternating high workload to the CPU and to the GPU.” Conroy, [0196]. Again, it is only common sense that the processor cores would receive substantially all of the power when the workload is devoid of graphics, and the graphics controller would receive substantially all of the power when the workload is graphics-heavy.

127. Accordingly, it is my opinion that a POSITA would have expected a multi-domain processing system like Finkelstein’s to alternate between executing processor-heavy applications and graphics-heavy applications. Based on the

disclosures in Finkelstein and common knowledge about workloads in a typical computing system like Finkelstein's, a POSITA would have understood that its power management logic would allocate substantially all of the power budget E^k to the processor core power plane (first domain) for a processor-heavy workload and substantially all of the power budget to the graphics controller (second domain) for a graphics-heavy workload. *See, e.g.*, Finkelstein [0026], [0034].

128. I have also been asked to consider whether Finkelstein would have rendered it obvious to allocate substantially all of the variable power budget to the graphics controller (second domain) for a second workload submitted to the system after a first workload. In my opinion, this would have been obvious in order to efficiently allocate power to the processor core power plane (first domain) for processor-heavy workloads and to efficiently allocate power to the graphics controller power plane (second domain) for graphics-heavy workloads, as was known in the art. *See e.g.*, Gwennap, 6 (“If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs.”); Conroy [0196] (“Various applications provide various workloads to each of the subsystems. For example, program development tools and scientific applications present a high load

to the CPU, but almost no load to the GPU that leads to an asymmetric load profile of the system (e.g. the CPU consumes a lot more power than the GPU). Many professional applications present an alternating high workload to the CPU and to the GPU that results in an alternating asymmetric load profile of the system. Advanced user interfaces or graphics editing application present a high load to the GPU and a modest load to the CPU that leads to another asymmetric load profile to the system.”).

129. A POSITA would have reasonably expected to succeed in configuring Finkelstein’s power management logic to do so because it would have required no more than routine skill in the art. For example, because power distribution preferences may be “user (or application defined)” (Finkelstein, [0034]), a POSITA would have known to program the power management logic to allocate substantially all of the available power to the processor core power plane (first domain) for first workload when it contains a processor-heavy workload with little or no graphics operations to be performed. Likewise, a POSITA would have known to program the power management logic to allocate substantially all of the available power to the graphics controller (second domain) when the workload during a subsequent interval is graphics-heavy and requires little or no support from the processor cores. *See* Finkelstein, [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more

Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”). Doing so would have involved known techniques, e.g., writing software of the kind those ordinarily skilled in the art would have been able to write. *See, e.g.*, Therien, [0014] (“[A]ny of the methods described herein may be performed by hardware, software (including lower level code, such as microcode), or a combination of hardware and software.”), Conroy, [0191] (“[T]he techniques may be carried out in a computer system or other data processing system in response to its processor, such as a microprocessor or a microcontroller, executing sequences of instructions contained in a memory”), [0195] (“The power used by at least a subset of each of the subsystems is controlled, e.g., by a microcontroller”), [0196] (“[T]he load profile may be identified using workloads determined by measuring/sensing power (e.g. current drawn) by each subsystem or by measuring power for certain subsystems and estimating or predicting power for other subsystems or by estimating power for all subsystems. In another embodiment, the load profile may be identified using workloads determined by the operating system out from historical scheduling data. In yet another embodiment, to identify the load profile of the system, the information about the workload of the subsystem provided by an application may be used.”).

130. Accordingly, it is my opinion that a POSITA would have found it obvious to dynamically allocate substantially all of Finkelstein’s variable power

budget (E_n^k) to the processor core power plane (first domain) for a first (processor-heavy) workload, and to dynamically allocate substantially all of Finkelstein's variable power budget (E_{n+1}^k) to the graphics controller power plane (second domain) for a second (graphics-heavy) workload executed after the first (processor-heavy) workload.

3. Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.”

131. In my opinion, Finkelstein at least suggests the additional features of claim 17 because its power management logic allocates the “entire budget” to a “single power plane” when needed, based on “user (or application defined) *preferences*.” Finkelstein, [0034] (emphasis added). A POSITA would have understood this to be true when the processor cores (first domain) and graphics controller (second domain) share a common package budget. For example, Finkelstein explains that “it is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.” Finkelstein, [0026].

132. Finkelstein discloses that the graphics controller (second domain), which performs “Graphics Effect(s) (GFX),” may demand more power relative to

the processor cores (first domain) during a “GFX intensive workload.” Finkelstein, [0026]. Accordingly, a POSITA would have understood Finkelstein to disclose distributing more power to the graphics controller during graphics-heavy workloads. *See* Finkelstein, [0026] (teaching to “allow for use of . . . unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload”), [0050] (describing GFX functions as “high-performance”). During time intervals where such loads are present, a POSITA would have understood to prioritize these “high-performance” graphics workloads over workloads performed by the processor core (first domain) to achieve efficient graphics performance.

133. It was well known in the art to prioritize a graphics domain over other domains. *See* Gwennap, 6 (“If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs.”); Therien, [0017] (“[A] computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU.”); Bose, Abstract (“During the first mode, instructions associated with applications are executed by a first asymmetric core, and a second asymmetric core is inactive. During the second mode, the instructions are executed by the second asymmetric core, and the first

asymmetric core is inactive.”), [0010] (disclosing “reassigning threads of the workload to cores remaining active by assigning threads based on priority constraints and thread execution history to improve the operational efficiency of the multi-core architecture”), [0021] (“[C]ore activation/ deactivation may be provided based upon hardware or priority constraints.”), [0085] (describing a “thread scheduler” that “priorit[izes]” and “assigns” power distribution to “high performance threads” over “low performance” threads).

134. Accordingly, it is my opinion that a POSITA would have understood when Finkelstein’s “user (or application defined) preferences” indicate a preference for the graphics controller power plane (second domain) over the processor core power plane (first domain) during a given time interval (such that $W_{k2} > W_{k1}$), Finkelstein’s power management logic would favor power allocation to the second domain over the first domain. Finkelstein, [0034].

135. I have also been asked to consider whether a POSITA would have found it obvious to allocate Finkelstein’s available variable power budget to the graphics controller (second domain) according to a preference value that favors the graphics controller (second domain) over the processor cores (first domain). In my opinion, the answer is yes. When a graphics-heavy workload is presented to Finkelstein’s computing system, a POSITA would have been motivated to set a preference value (e.g., to set $W_{k2} > W_{k1}$, such that W_{k2} represents a preference value)

that favors power allocation to Finkelstein's graphics controller (second domain) over the processor cores (first domain) to ensure the graphics operations are performed quickly and efficiently. Again, this was well known in the art. *See* Gwennap, 6 ("If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs."); Therien, [0017] ("[A] computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU."); Bose, Abstract ("During the first mode, instructions associated with applications are executed by a first asymmetric core, and a second asymmetric core is inactive. During the second mode, the instructions are executed by the second asymmetric core, and the first asymmetric core is inactive."), [0010] (disclosing "reassigning threads of the workload to cores remaining active by assigning threads based on priority constraints and thread execution history to improve the operational efficiency of the multi-core architecture"), [0021] ("[C]ore activation/ deactivation may be provided based upon hardware or priority constraints."), [0085] (describing a "thread scheduler" that "priorit[izes]" and "assigns" power distribution to "high performance threads" over "low performance" threads).

136. It is also my opinion that a POSITA would have reasonably expected to succeed in configuring Finkelstein's power management logic to allocate the variable power budget according to a preference value that favors Finkelstein's graphics controller (second domain) over the processor cores (first domain) because it would have required no more than routine skill in the art to do so. For example, because the power distribution preferences may be "user (or application) defined," a POSITA would have known to program power management logic to favor the second domain over the first domain (e.g., by ensuring that $W_{k2} > W_{k1}$ for graphics-heavy workloads). Again, doing so would have involved known techniques, e.g., writing software of the kind those ordinarily skilled in the art would have been able to write. *See, e.g.*, Therien, [0014] ("[A]ny of the methods described herein may be performed by hardware, software (including lower level code, such as microcode), or a combination of hardware and software."), Conroy, [0191] ("[T]he techniques may be carried out in a computer system or other data processing system in response to its processor, such as a microprocessor or a microcontroller, executing sequences of instructions contained in a memory"), [0195] ("The power used by at least a subset of each of the subsystems is controlled, e.g., by a microcontroller"), [0196] ("[T]he load profile may be identified using workloads determined by measuring/sensing power (e.g. current drawn) by each subsystem or by measuring power for certain subsystems and estimating or predicting power for other

subsystems or by estimating power for all subsystems. In another embodiment, the load profile may be identified using workloads determined by the operating system out from historical scheduling data. In yet another embodiment, to identify the load profile of the system, the information about the workload of the subsystem provided by an application may be used.”).

C. Ground 3: Claims 8 and 12-17 Would Have Been Obvious over Finkelstein and Felter

137. I have also been asked to consider whether claims 8 and 12-17 would have been obvious based on the combined disclosures of Finkelstein and Felter. In my opinion, a POSITA would have found claims 8 and 12-17 obvious based on such their combined teachings, as I further explain below.

1. Overview of the Finkelstein-Felter Combination

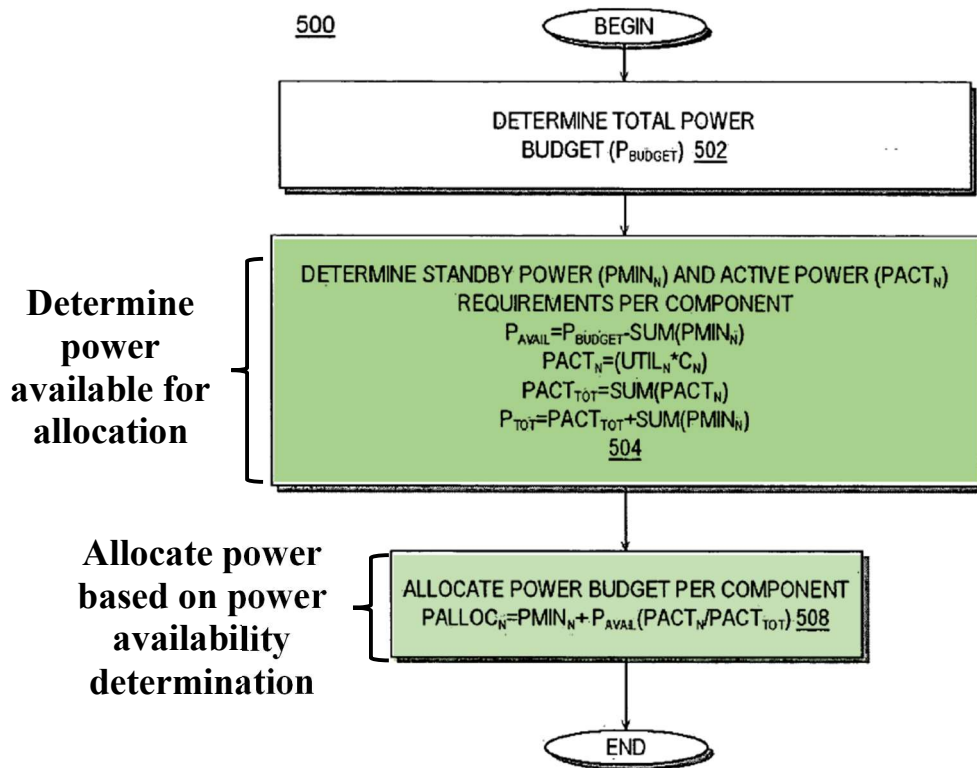
138. Felter’s data processing system 100 and Finkelstein’s computing system 100 have a lot in common. Both include multiple processors 102-N connected by a shared bus 104. *Compare* Felter, [0026], FIG. 1, *with* Finkelstein, [0010], FIG. 1. Both are coupled to a system memory (e.g., DRAM). *Compare* Felter, [0042], FIG. 1, *with* Finkelstein, [0013], [0043], FIGS. 1, 6. Both references center on logic that dynamically allocates power among various domains, including domains associated with different processor cores. *Compare* Felter, [0030], [0042]-[0049], [0057]-[0061], [0064]-[0066], FIGS. 2, 5, *with* Finkelstein, [0016], [0023]-

[0024], [0031]-[0035], [0039]-[0040], [0042], FIGS. 1, 5. And both use activity / power monitors to aid in doing so. *Compare* Felter, [0029]-[0031], [0050]-[0051], [0059], [0065]-[0066], FIGS. 2, 4-5, *with* Finkelstein, [0016], [0019], [0023]-[0024], [0042], [0049], FIGS. 1, 5.

139. Felter's method of allocating power and Finkelstein's method of power management also have a lot in common. *Compare* Felter, [0057]-[0066], FIG. 5, *with* Finkelstein, [0027]-[0040], FIG. 5. The only notable difference between Finkelstein's power allocation equation (*see* Finkelstein, [0034]) and Felter's power allocation equation (*see* Felter, [0061])—both of which are linear—is that Felter expressly accounts for “standby values (minimum power values) of each system component,” Felter, [0058], whereas Finkelstein does not do so explicitly.³ Felter

³ Although Finkelstein allocates an energy budget and Felter allocates a power budget, a POSITA would have understood the two to be equivalent. A POSITA would have understood a power budget can be converted to an energy budget, and vice versa, simply by multiplying the power budget by the time interval (or dividing the energy budget by the time interval). *See* Finkelstein, [0028] (“ $E_{n+1} = \alpha E_n + (TDP_n - P_n)\Delta t_n$ ”). The '316 patent itself also makes clear that the claimed “power budget” may be expressed in terms of energy (using Joules), not just in terms of power (using

refers to this as “a modified pro rata technique,” Felter, [0061], whereas Finkelstein’s method of power allocation merely employs a pro rata power allocation. *See* Finkelstein, [0034] (disclosing formula (2): “ $E^{k,i}=W_{ki}E^k$ ”).



Felter, FIG. 5 (annotated)

Watts). *See* '316 patent, 3:22-62 (describing equation [1]), claim 7 (defining “the power budget for the current time interval” as an energy “ E_n ” and saying “to determine the power budget” in terms of “energy”). A POSITA thus would have understood the claimed “power budget” refers interchangeably to power (in Watts) and energy (in Joules).

140. In Felter's modified pro rata technique, which I have annotated above, each system component (domain) not only receives a share of dynamically allocated power (P_{AVAIL}) but also a "minimum operational power" (also called "standby power"), denoted $PMIN_N$. See Felter, [0039], [0042]-[0049], [0058], [0061], [0065], FIG. 5. Felter's power administrator 200 (power controller) determines "the power budget and the active and *standby demands* of each of the components" and then allocates, to each component, "the component's standby power requirements [$PMIN_N$] (which must always be met) and a portion of the available power P_{AVAIL} ." Felter, [0061], FIG. 5.

141. According to Felter, the modified pro rata technique "effectively distributes the available system power to the system components in a manner that (1) ensures that each component receives a minimum level of power and (2) allocates the available active power in proportion to the requirement for the components by the workload(s) executing on the system." Felter, [0061]. Moreover, Felter explains that including "a reservoir for minimal component activity" in the standby power allocated to each component "ensure[s] there is always enough power for each component to obtain some minimal work from it." As I further explain below, a POSITA would have found it obvious for the same reasons to adapt Finkelstein's power management logic to perform Felter's "modified pro rata technique" instead of Finkelstein's pro rata power allocation technique.

2. A POSITA Would Have Found It Obvious to Adapt Finkelstein's Power Management Logic to Perform Felter's "Modified Pro Rata Technique" for Power Allocation

142. Again, Finkelstein and Felter describe similar techniques for managing power in similar systems. For example, they define their respective power/energy budgets in the same way. Felter discloses an “approach [that] relies on the not unreasonable assumption that very recent activity is a good predictor of current activity,” which “is also desirable for its responsiveness in the context of a rapidly changing operating environment.” Felter, [0051]. For example, Felter teaches that “the predicted activity for time period N+1 is a simple function of the monitored activity for time period N.” Felter, [0051]. Likewise, Finkelstein discloses an “energy budget defined iteratively as follows: $E_{n+1} = \alpha E_n + (TDP_n - P_n) \Delta t_n$.” Finkelstein, [0028]-[0029]. As this equation makes clear, the budget for time period N+1 (E_{n+1}) is a simple function of the monitored budget for time period N (E_n), just as Felter discloses.

143. Additionally, both Finkelstein and Felter allocate their respective power/energy budgets in similar ways. As I explain above for claim 13 in the context of Ground 2, Finkelstein allocates power on a pro rata basis using formula (2):

$$E^{k,i} = W_{ki} E^k \quad (\text{Eq. 2})$$

where E^k is the power budget, and the W_{ki} values are “user (or application defined) preferences.” Finkelstein, [0034]. Felter similarly allocates power “us[ing] a

modified pro rata technique,” as I explain above for claim 8 in the context of Ground 1. Felter, [0061] (emphasis added). Similar to Finkelstein’s pro rata power allocation, Felter allocates power based on preferences (realized via the $PACT_N$ values), but only after first reserving a “minimum power” ($PMIN_N$) for each domain N:

$$PALLOC_N = PMIN_N + (PACT_N/PACT_{TOT}) (PBUDGET - SUM(PMIN_N)),$$

where $PMIN_N$ are the minimum powers, $PBUDGET$ is the power budget, and $PACT_N/PACT_{TOT}$ allocates power based on certain preferences. Felter, [0057]-[0061], FIG. 5.

144. Again, Felter explains that this “modified pro rata technique . . . effectively distributes the available system power to the system components in a manner that (1) ensures that each component receives a minimum level of power and (2) allocates the available active power in proportion to the requirement for the components by the workload(s) executing on the system.” Felter, [0061]. Felter also explains that including “a reservoir for minimal component activity . . . in the ‘standby’ power” (i.e., $PMIN_N$ values) will “ensure there is always enough power for each component to obtain some minimal work from it.” Felter, [0065].

145. Looking at Felter’s disclosures and applying them to Finkelstein’s similar disclosures, a POSITA would have found it obvious to adapt Finkelstein’s power management logic so that it applies Felter’s modified pro rata technique rather

than Finkelstein's slightly simpler pro rata power allocation technique. To be more specific, it is my opinion that a POSITA would have found it obvious to adapt Finkelstein's formula (2) by applying Felter's power allocation equations summarized in Felter's Figure 5. The result would have been straight-forward:

$$E^{k,i} = E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i}) \quad (\text{Eq. 2, modified})$$

where $E_{min}^{k,i}$ are the minimum powers, E^k is the power budget, and $W_{k,i}$ allocates power based on predicted activity, as Felter instructs. *Compare* Finkelstein, [0028]-[0029] (explaining how to define a variable power/energy budget as a simple function of the previous power/energy budget), [0034] (explaining how to allocate the variable power/energy budget using a pro rata technique), *with* Felter, [0051] (explaining how to define a variable power/energy budget as a simple function of the previous power/energy budget), [0057]-[0061] (explaining how to allocate the variable power/energy budget using a *modified* pro rata technique).

146. In my opinion, a POSITA would have been motivated to include a minimum power for each power plane i (which I have designated as $E_{min}^{k,i}$) to "ensure there is always enough power for each component to obtain some minimal work from it." Felter, [0065]. A POSITA would have recognized that such an approach would further Finkelstein's goal to support "smooth operation of critical applications." Finkelstein, [0024].

147. Additionally, it is my opinion that a POSITA would have desired to configure Finkelstein's power management logic to allocate the available active power according to "the workload(s) executing on the system." Felter, [0061]. For example, one of Finkelstein's goals is "to share a common package power/energy budget and allow for use of . . . unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized." Finkelstein, [0026]. A POSITA would have understood that Felter's modified pro rata technique furthers this goal by "distribut[ing] scarce system power equitably so that the system performance of each component is impacted proportionately." Felter, [0061]. Thus, in my opinion, a POSITA would have found Felter's modified pro rata power allocation technique to be advantageous when applied to Finkelstein's disclosures.

148. Moreover, Finkelstein discloses "user (or application defined) preferences that describe how budget E^k is distributed among power planes." Finkelstein, [0034]. Finkelstein does not provide any specific examples of how such preferences would be set, but a POSITA would have understood that Felter discloses an example of how to do so. For instance, Felter explains that "the active power $[PACT_N]$ predicted for a component N is indicated as the product of the activity (UTIL) predicted for component N ($UTIL_N$) and a constant of proportionality for component N (C_N)." Felter, [0059]; *see also* Felter, FIG. 5 (" $PACT_N = (UTIL_N * C_N)$ ").

A POSITA would have found this approach to be both compatible with Finkelstein's W_{ki} values and to further Finkelstein's goals.

149. Regarding the C_N values, a POSITA would have found it obvious for either a user or an application to set those values based on preferences that account for the well-known tradeoffs between efficiency and performance. For example, a larger value of C_N would favor performance by allotting more power to domain N, whereas a smaller value of C_N would favor efficiency by allotting less power to domain N. Regarding the $UTIL_N$ values, which represent utilization, a POSITA would have understood that each application dictates those values based on how they are being used. *See* Finkelstein, [0026] (explaining the graphics domain may require more power under “a GFX intensive workload”); Gwennap, 6 (“If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs.”); Conroy [0196] (“Many professional applications present an alternating high workload to the CPU and to the GPU that results in an alternating asymmetric load profile of the system.”). In my opinion, therefore, a POSITA would have found it obvious to set Finkelstein's “user (or application defined) preferences” (the $W_{k,i}$ values) by applying Felter's teachings regarding $PACT_N$. This only

becomes more apparent when noting Finkelstein's $W_{k,i}$ values are mathematically equivalent to Felter's ratios $PACT_N/PACT_{TOT}$ in the combination I am contemplating, because both values represent the weighting factor for allocating the power after allocating the minimum power to each power domain. *Compare* Felter, [0061] ($PALLOC_N = P_{MIN_N} + P_{AVAIL} * (PACT_N/PACT_{TOT})$), with Finkelstein, [0034] ($E^{k,i} = W_{ki}E^k$). Since the values of $PACT_N/PACT_{TOT}$ are set by $UTIL_N$ and C_N , a POSITA would have realized Finkelstein's preference values $W_{k,i}$ can be set in the same way to achieve the same benefits in the combination of Finkelstein and Felter. *See* Felter, [0059], FIG. 5.

150. When adapting Finkelstein's power management logic to perform Felter's modified pro rata power allocation technique, a POSITA would have reasonably expected to succeed because such a modification would not have required anything more than routine and ordinary skill in the art. As I mention in the paragraph above, the $W_{k,i}$ and $PACT_N/PACT_{TOT}$ values are mathematically equivalent in the combination I am contemplating. As I also describe above, Felter's method of determining $PACT_N$ (as defined by the equation $PACT_N = UTIL_N * C_N$) provides an exemplary way of to define Finkelstein's "user (or application defined) preferences" $W_{k,i}$. *See* Finkelstein, [0034]; Felter, [0059], FIG. 5. A POSITA therefore would have been readily able to apply Felter's modified pro rata technique to set the $W_{k,i}$ values in the Finkelstein-Felter combination. The same is true of the

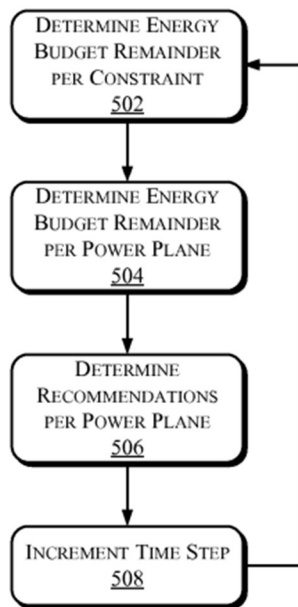
minimum power values ($E_{\min}^{k,i}$), which would not have required anything more than routine and ordinary skill in the art to set. For example, a POSITA would have understood how to acquire the minimum power values ($E_{\min}^{k,i}$) through manufacturer guidelines, routine observation of each power plane's power consumption, or discrete settings, all of which were well known in the art. *See* Finkelstein, [0034] (“For constraint k, user (or application defined) preferences that describe how budget E^k is distributed among power planes may be determined.”); Bose, [0048] (“Discrete settings of percentages . . . for core activity is also a possible implementation / embodiment.”), [0055] (explaining that component utilization “may be based on the design of the chip or core or based on performance metrics of the like”). In my opinion, therefore, a POSITA would have reasonably expected to succeed when adapting Finkelstein's power allocation logic to perform Felter's modified pro rata technique rather than Finkelstein's pro rata power allocation technique.

3. Independent claim 8

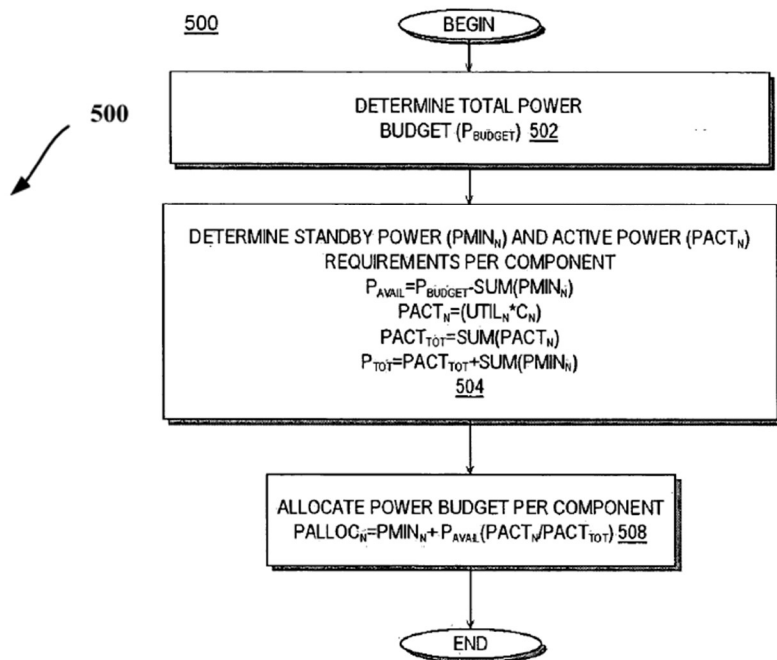
a. [8pre]: “A method comprising:”

151. The combination of Finkelstein and Felter includes feature [8pre]. As I explain in Sections IX.C.1-2 above, the combination of Finkelstein and Felter performs a method for allocating power in a multi-processor system. Finkelstein, for example, discloses a method for “provid[ing] efficient power management for multiple processor cores.” Finkelstein, [0009]; *see also* Finkelstein, FIG. 5. Felter

discloses a similar “method for managing power in a data processing system having multiple components.” Felter, Abstract. More specifically, Felter discloses “a method for allocating power budgets to the system components based on expected levels of activity.” Felter, [0016]; *see also* Felter, [0057]-[0061], FIG. 5 (below, right).



Finkelstein, FIG. 5



Felter, FIG. 5

152. In the combination of Finkelstein and Felter I have considered and described in preceding Sections IX.C.1-2, a POSITA would have found it obvious to adapt Finkelstein’s power management logic to perform a method of power allocation using Felter’s modified pro rata technique. This constitutes a “method” as recited in feature [8pre].

- b. **[8a]: “determining, in a power controller of a multi-domain processor, a power budget for the multi-domain processor for a current time interval, the multi-domain processor including at least a first domain and a second domain”**

153. The combination of Finkelstein and Felter includes feature [8a]. Finkelstein discloses a “multiple core processor” (multi-domain processor) with “one or more processor cores” operating on “multiple power planes” (at least first and second domains). Finkelstein, [0009]; *see also* Finkelstein, FIG. 1. Finkelstein further discloses a “power management logic” (a power controller of the multi-domain processor) for “defin[ing]” and “manag[ing]” a “controller budget” (which Finkelstein also refers to as a “power/energy budget”). Finkelstein, [0016], [0026], [0028], [0039]. Finkelstein’s Figure 1, which I have reproduced below with highlights, shows an exemplary multi-processor computing system 100. *See* Finkelstein, [0010], FIG. 1. This is the same computing system Finkelstein discloses, although it would be adapted to incorporate Felter’s modified pro rata power allocation technique, as I describe above.

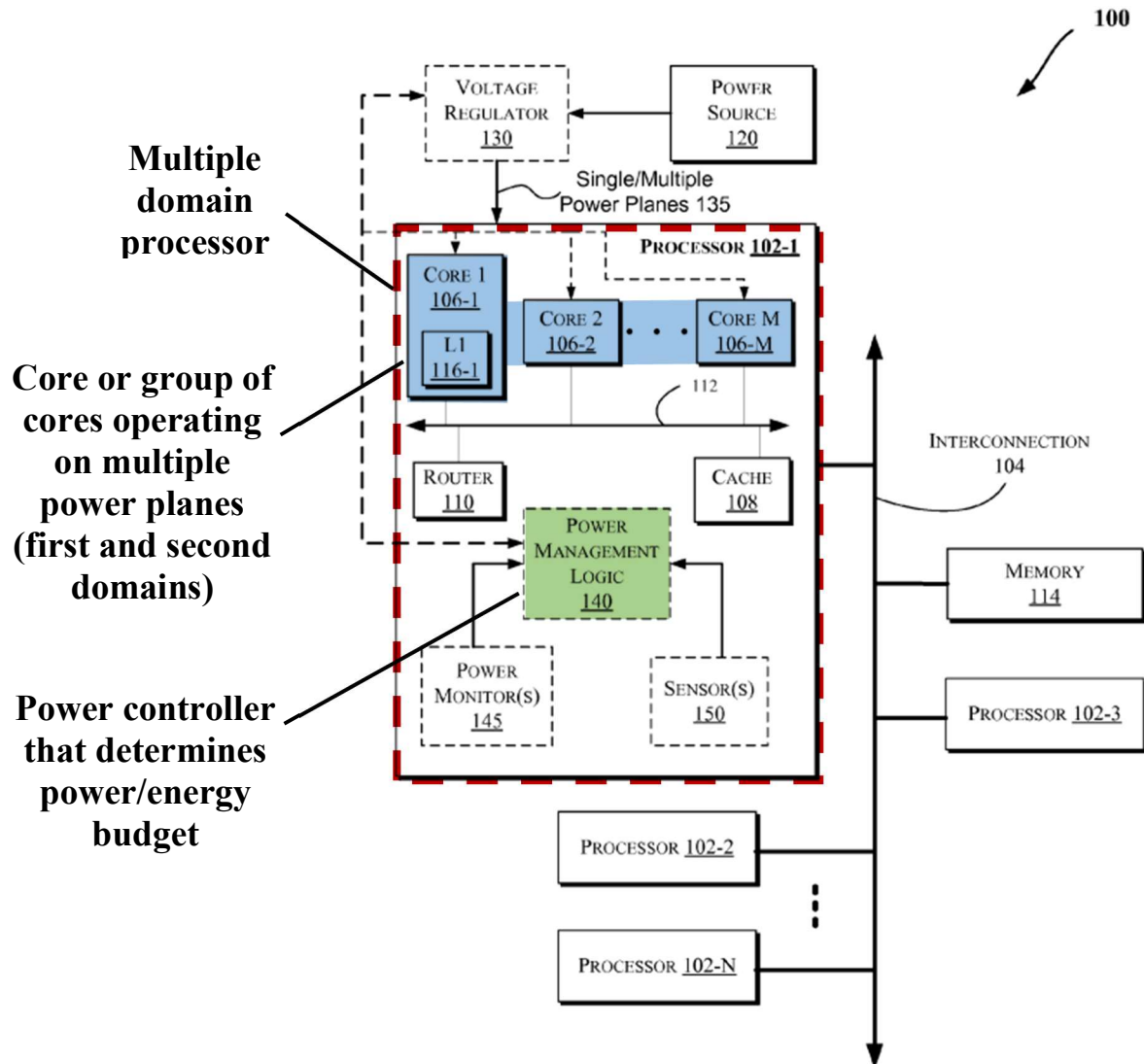


FIG. 1

Finkelstein, FIG. 1 (annotated)

154. As I explain in Sections IX.C.1-2 above, the combination of Finkelstein and Felter would follow Finkelstein's basic architecture. Each processor 102 (e.g., processor 102-1) has multiple cores 106-1 through 106-M and power management logic 140 that cooperates with one or more voltage regulator(s) (VR) 130 to manage

power in the system. *See* Finkelstein, [0011] (“[T]he processor 102-1 may include one or more processor cores 106-1 through 106-M . . .”), [0016] (“The logic 140 may instruct the VR 130, power source 120, and/or individual components of system 100 (such as the cores 106) to modify their operations.”), FIG. 1. The voltage regulators 130, in turn, are “coupled to a single power plane 135 (e.g., supplying power to all the cores 106) or to *multiple power planes 135 (e.g., where each power plane may supply power to a different core or group of cores)*.” Finkelstein, [0014] (emphasis added); *see also* Finkelstein, FIG. 1 (showing the same).

155. Finkelstein’s power management logic determines the power/energy budget for each interval and then decides how to allocate it to the power planes that share the budget. *See* Finkelstein, [0039] (describing the basic process), FIG. 5 (illustrating the same). Then, after allocating power to the various processor planes, Finkelstein’s power management logic applies “DVS, pure throttling techniques, or some combinations thereof to control the power consumption of processors 102.” Finkelstein, [0024]. A POSITA thus would have understood each of Finkelstein’s power planes corresponds to a domain. This is consistent with the ’316 patent, which uses the term domain “to mean a collection of hardware and/or logic that operates at the same voltage and frequency point.” ’316 patent, 1:52-56. This remains true in the combination of Finkelstein and Felter. Finkelstein’s power management logic is merely adapted to perform a slightly different algorithm, as I explain above.

156. According to Finkelstein, “power setting(s) (e.g., voltage and/or frequency changes) may be made according[] to a current budget.” Finkelstein, [0028]. Finkelstein’s power management logic (the power controller) defines the power/energy budgets⁴ (E_{n+1}^k) for each interval, and then it determines how to allocate each power/energy budget among the associated power planes (including the first and second domains, which share a budget). *See* Finkelstein, [0028]-[0039]. Again, in the combination of Finkelstein and Felter, a POSITA would have

⁴ Although Finkelstein allocates an energy budget and Felter allocates a power budget, a POSITA would have understood the two to be equivalent. A POSITA would have understood a power budget can be converted to an energy budget, and vice versa, simply by multiplying the power budget by the time interval (or dividing the energy budget by the time interval). *See* Finkelstein, [0028] (“ $E_{n+1} = \alpha E_n + (TDP_n - P_n) \Delta t_n$ ”). The ’316 patent itself also makes clear that the claimed “power budget” may be expressed in terms of energy (using joules), not just in terms of power (watts). *See* ’316 patent, 3:22-62 (describing equation [1]), claim 7 (defining “the power budget for the current time interval” as an energy “ E_n ” and saying “to determine the power budget” in terms of “energy”). A POSITA thus would have understood the claimed “power budget” refers interchangeably to power (in watts) and energy (in joules).

found it obvious to adapt Finkelstein's power management logic to perform this task by incorporating Felter's modified pro rata power allocation technique. *See* Sections IX.V.1-2 above.

157. In the basic flow diagram illustrated in Finkelstein's Figure 5, Finkelstein's computing system first determines "a current budget" (E_{n+1}^k), which is "defined iteratively" in terms of the previous interval (E_n^k). Finkelstein, [0028]-[0029] (providing formula (1): " $E_{n+1} = \alpha E_n + (TDP_n - P_n) \Delta t_n$ "); *see also* Finkelstein, [0039] ("[T]he energy budget remainder per constraint E_n^k may be determined (e.g., by the logic 140 in accordance with formula (1) above)"). Likewise, Felter's first step is to determine a power budget (P_{BUDGET}) for the upcoming interval. *See* Felter, [0057], FIG. 5 (block 502). Also similar to Finkelstein, Felter determines the power budget based on component activity levels (utilization) during the previous interval, which correspond to the power consumed by each component during the previous interval. *See* Felter, [0051] ("[T]he predicted activity for time period N+1 is a simple function of the monitored activity for time period N"), [0054] ("[P]ower administrator 200 includes allocating (block 306) power budgets to the various system components based on the predicted level of activity."), FIGS. 3, 5.

158. When determining the budget, Finkelstein's power management logic may receive information pertaining to "variations in temperature, operating frequency, operating voltage, power consumption, inter-core communication

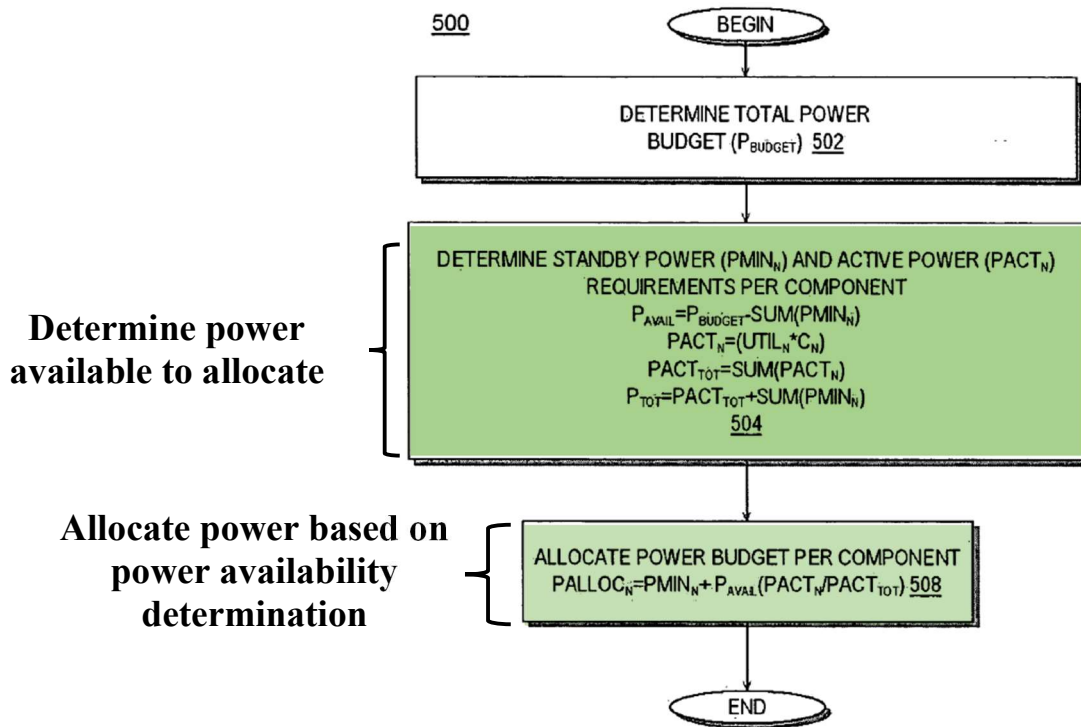
activity, etc.[] and/or information from one or more power monitoring logics 145” and “instruct the VR 130, power source 120, and/or individual components of system 100 (such as the cores 106) to modify their operations” based on the received information. Finkelstein, [0016]. Again, this matches Felter’s disclosures. *See* Felter, [0031]-[0033] (describing the use of performance monitors), [0050] (same), [0054] (explaining the power budget is “influenced by environmental or external conditions ... such as the ambient temperature”).

159. Accordingly, in the combination of Finkelstein and Felter, a power controller (Finkelstein’s power management logic, as adapted to perform Felter’s modified pro rata power allocation technique) of a multi-domain processor (Finkelstein’s processor 102-1) determines a power budget (E_{n+1}^k) for the multi-domain processor for a current time interval (denoted by the index $n+1$). Moreover, the multi-domain processor (Finkelstein’s processor 102-1) includes at least a first domain (a first processor core power plane 135) and a second domain (a second processor core power plane 135).

c. [8b]: “determining, in the power controller, a portion of the power budget to be allocated to the first and second domains,”

160. The combination of Finkelstein and Felter includes feature [8b]. As adapted, Finkelstein’s power management logic (power controller) employs Felter’s modified pro rata technique in order to allocate the power/energy budget to the

power planes that share the budget (including the first and second processor power planes). Felter, [0061], *see also* Felter, [0050]-[0066], FIGS. 3-5 (collectively describing key aspects of Felter's modified pro rata power allocation technique).



Felter, FIG. 5 (annotated)

161. As Figure 5 shows above (and as I explain above for feature [8b] in the context of Felter and Ground 1), Felter teaches that “the amount of power budgeted for each component is equal to the component’s standby power requirements (which must always be met) and a portion of the available power P_{AVAIL} .” Felter, [0061]. Felter refers to the portion of the power budget (P_{BUDGET}) to be allocated to the Nth domain as $PALLOC_N$. Felter, [0061], FIG. 5 (block 508).

162. In the combination of Finkelstein and Felter, a POSITA would have found it obvious to adapt Finkelstein's power management logic, as I describe above, so that it would apply Felter's modified pro rata technique. As I explain above, Finkelstein's formula (2) would be modified as follows to include "a reservoir for minimal component activity [$E_{min}^{k,i}$]" (Felter, [0065]):

$$E^{k,i} = E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i}) \quad (\text{Eq. 2, modified})$$

See Felter, [0061], FIG. 5 (blocks 504, 508); Finkelstein, [0034]; Sections IX.C.1-2 above.

163. According to Felter, this approach "effectively distributes the available system power to the system components in a manner that (1) ensures that each component receives a minimum level of power and (2) allocates the available active power in proportion to the requirement for the components by the workload(s) executing on the system." Felter, [0061].

164. Accordingly, the combination of Finkelstein and Felter would determine, in Finkelstein's modified power management logic (power controller), a first portion ($E^{k,1}$) of the power budget (E^k) to be allocated to the first processor

core(s) (first domain) and a second portion ($E^{k,2}$) of the power budget (E^k) to be allocated to the second processor core(s) (second domain).⁵

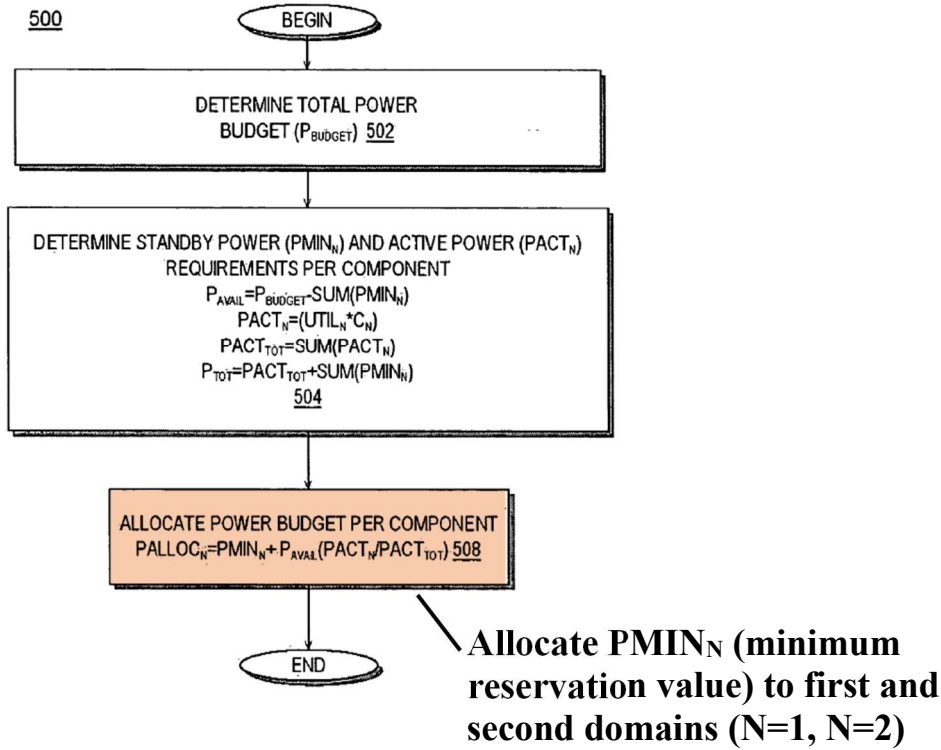
d. [8c]: “including allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain, ”

165. The combination of Finkelstein and Felter includes feature [8c] because, as modified, Finkelstein’s power management logic allocates, to each power plane, “a reservoir for minimal component activity” (a minimum reservation value), as taught by Felter. Felter, [0065]; *see also* Felter, [0061] (explaining that “the amount of power budgeted for each component” includes “the component’s standby power requirements (which must always be met)”), FIG. 5; Section IX.C.1-2

⁵ The sum of $E^{k,i}$ over all i power planes equals the total budget E^k , such that $E^{k,i}$ for each power plane i corresponds to a portion of the total power budget. The math is fairly straight-forward. According to Felter’s modified pro rata power allocation technique, $PACT_{TOT} = SUM(PACT_N)$. Felter, FIG. 5 (block 504). It follows that $SUM(PACT_N/PACT_{TOT}) = SUM(PACT_N)/PACT_{TOT}$ (since $PACT_{TOT}$ is a constant) $= PACT_{TOT}/PACT_{TOT} = 1$. Since $W_{k,i}$ corresponds to $PACT_N/PACT_{TOT}$ in the combination of Finkelstein and Felter, this means $\sum_i W_{k,i} = 1$. Accordingly, it follows that $\sum_i E_{k,i} = \sum_i [E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i})] = \sum_i E_{min}^{k,i} + E^k \cdot \sum_i W_{k,i} - \sum_i E_{min}^{k,i} \cdot \sum_i W_{k,i} = \sum_i E_{min}^{k,i} + E^k - \sum_i E_{min}^{k,i} = E^k$.

above (describing the combination of Finkelstein and Felter), Section IX.A.1.d above (describing how Felter discloses feature [8c]).

166. As I explain in the context of Ground 1, Felter allocates to each domain a “fixed” $PMIN_N$ (also called “standby power”) corresponding to “the minimum operational power of the processor at its lowest operating point.” Felter, [0039] (explaining that “ $P_{proc}=G(f)+P_{min}$ where $G(f)$ is a polynomial in f [frequency] . . . , [and] P_{min} is the minimum operational power of the processor at its lowest operating point”), [0042]-[0049] (showing that each power allocation for “two independent frequency-scalable processors” includes the respective P_{min} value of each), [0058] (referring to the P_{min} values as “the standby values (minimum power values) of each system component”), [0065] (“reservoir for minimal component activity”), FIG. 5. More specifically, Felter instructs that its power administrator (analogous to Finkelstein’s power management logic) first determines “the power budget and the active and standby demands of each of the components” (*see* Felter, [0058]-[0060], FIG. 5 (block 504)), and then it allocates, to each component, its “standby power requirements [$PMIN_N$] (which must always be met) and a portion of the available power P_{AVAIL} .” Felter, [0061], FIG. 5 (block 508).



Felter, FIG. 5 (annotated)

167. In the combination of Finkelstein and Felter, where Finkelstein's power management logic applies Felter's modified pro rata technique, Finkelstein's equation (2) is modified as follows:

$$E^{k,i} = E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i}) \quad (\text{Eq. 2, modified})$$

where $E_{min}^{k,i}$ corresponds to Felter's minimum power / standby power P_{MIN_N} . See Section IX.C.1-2 above (describing the combination of Finkelstein and Felter). Therefore, it is my opinion that a POSITA would have found it obvious that the management logic, as modified, allocates a first minimum reservation value ($E_{min}^{k,1}$)

to the first processor core power plane (first domain) and a second minimum reservation value ($E_{\min}^{k,2}$) to the second processor core power plane (second domain).

- e. **[8d]: “and sharing a remaining portion of the power budget according to a first sharing policy value for the first domain and a second sharing policy value for the second domain; and”**

168. The combination of Finkelstein and Felter includes feature [8d] because “the amount of power budgeted for each component is equal to the component’s standby power requirements $[P_{\min N}]$ (which must always be met) and a portion $[P_{\text{ACT}N}/P_{\text{ACTTOT}}]$ of the available power P_{AVAIL} ,” as taught by Felter. Felter, [0061]; *see also* Felter, [0058]-[0059]; Section IX.C.1-2 above (describing the combination of Finkelstein and Felter), Section IX.A.1.e (explaining how Felter satisfies feature [8d]).

169. As I explain above, Felter’s power administrator (analogous to Finkelstein’s power management logic) allocates to each component “a reservoir for minimal component activity.” Felter, [0065]; *see also* Felter [0039], [0042]-[0049], [0058], [0061]. Felter’s power administrator then determines what remains of the power budget, a quantity Felter refers to as P_{AVAIL} . According to Felter, “an available power figure (P_{AVAIL}) is determined by subtracting from the maximum power deliverable by the system the sum of all the standby values.” Felter, [0058]; *see also* Felter, FIG. 5 (block 504) (“ $P_{\text{AVAIL}} = P_{\text{BUDGET}} - \text{SUM}(P_{\min N})$ ”). This quantity

(P_{AVAIL}) is the “remaining portion of the power budget,” as claimed, after allocating “the standby values (minimum power values) of each system component.” Felter, [0058].

170. In the combination of Finkelstein and Felter, where Finkelstein’s power management logic applies Felter’s modified pro rata technique, Finkelstein’s equation (2) is modified as follows:

$$E^{k,i} = E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i}) \quad (\text{Eq. 2, modified})$$

where the term in parentheses corresponds to Felter’s P_{AVAIL} (the remainder of the power budget after allocating the minimum power values to each power plane), and $W_{k,i}$ corresponds to Felter’s PACT_N/PACT_{TOT} values (the relative proportions of the remaining power budget allocated to each power plane). *See* Section IX.C.1-2 above (describing the combination of Finkelstein and Felter).

171. Felter explains that “P_{AVAIL}=P_{BUDGET} – SUM(PMIN_N).” Felter, FIG. 5. Again, it is what remains of the power budget (P_{BUDGET}) after allocating the minimum power values (PMIN_N) to each component N. Just like P_{AVAIL}, the expression $(E^k - \sum_i E_{min}^{k,i})$ in modified equation 2 above is what remains of the power budget (E^k) in the Finkelstein-Felter combination after allocating the minimum power values ($E_{min}^{k,i}$) to each power plane i. Felter’s PACT_N/PACT_{TOT} values and Finkelstein’s $W_{k,i}$ values are similarly related to one another. Just like

Felter's $PACT_N/PACT_{TOT}$ values, Finkelstein's $W_{k,i}$ values are the claimed "sharing policy value[s]" for each power plane (domain) i . *See* Section IX.A.1.e above (discussing Ground 2, feature [13d]). Thus, as modified, Finkelstein's power management logic "allocates the available active power $[E^k - \sum_i E_{min}^{k,i}]$ in proportion to the requirement for the components by the workload(s) executing on the system." Felter [0061]; *see also* Felter, FIG. 5.

172. As I explain above, a POSITA would have looked to the way in which Felter sets its $PACT_N$ values to determine the $W_{k,i}$ values in the Finkelstein-Felter combination. *See* Section IX.C.2 above (describing the combination of Finkelstein and Felter). According to Felter, "the active power [$PACT_N$] predicted for a component N is indicated as the product of the activity (UTIL) predicted for component N ($UTIL_N$) and a constant of proportionality for component N (C_N)."
Felter, [0059]; *see also* Felter, FIG. 5 (" $PACT_N=(UTIL_N*C_N)$ "). Additionally, Felter's value $PACT_{TOT}$ is just the sum of all $PACT_N$. *See* Felter, [0060] (" $(PACT_{TOT})$ is computed as the sum of the individual component's expected active power."), FIG. 5 (" $PACT_{TOT}=SUM(PACT_N)$ "). Therefore, like the $PACT_N$ values themselves, Felter's ratios $PACT_N/PACT_{TOT}$ are also related to the expected utilization ($UTIL_N$) by a constant of proportionality (C_N). *See* Felter, [0059]-[0061], FIG. 5. The same would be true for the analogous $W_{k,i}$ values and power planes (domains) i in the combination of Finkelstein and Felter combination. For example,

a POSITA would have recognized from Felter's teachings that Finkelstein's $W_{k,i}$ may be set using a constant of proportionality ($C_{k,i}$) for each component and then and multiplying it by the associated utilization ($UTIL_i$) to get the equivalent of Felter's $PACT_i$ for each domain i and the equivalent total active power $PACT_{TOT} = \sum_i PACT_i$.

173. It is therefore my opinion that a POSITA would have found it obvious based on the combination of Finkelstein and Felter that Finkelstein's power management logic, as modified, would share a remaining portion ($E^k - \sum_i E_{min}^{k,i}$) of the power budget (E^k) according to a first sharing policy value ($W_{k,1}$) for the first domain (power plane, e.g., $i=1$) and a second sharing policy value ($W_{k,2}$) for the second domain (power plane, e.g., $i=2$).

f. [8e]: "controlling a frequency of the first domain and a frequency of the second domain based on the allocated portions."

174. The combination of Finkelstein and Felter includes feature [8e] because controlling frequency is one of the primary ways power management logic, like the type at issue here, enforces power management policies. As Finkelstein explains, "power setting(s) (e.g., voltage and/or frequency changes) may be made accordingly to a current budget." Finkelstein, [0028]; *see also* Finkelstein, [0039] ("At an operation 506, controller recommendations per power plane u_n^i , may be determined (e.g., by the logic 140 in accordance with formula (3) above)."). For example, "logic 140 may request the cores 106 to *modify their operating frequency*, power

consumption, etc.,” by e.g., implementing “frequency throttling” “according[] to a current [power/energy] budget.” Finkelstein, [0024], [0028]. Finkelstein also makes clear that such throttling “may be applied to multiple power plane processors.” Finkelstein, [0041].

175. A POSITA would have understood that adapting Finkelstein’s power management logic to perform Felter’s modified pro rata power allocation technique does not change this basic aspect of Finkelstein’s power management scheme. Felter’s power administrator 200 serves the same purpose. Just like Finkelstein, Felter uses power administrator 200 to perform “voltage-frequency scaling” (DVFS). Felter, [0035]-[0036]; *see also* Felter, [0037]-[0049] (describing voltage-frequency scaling for “two independent frequency-scalable processors using a single memory (DRAM) system”), [0065] (further describing “the voltage-frequency throttling described above”). Thus, regardless of whether Finkelstein’s power management logic performs its typical pro rata power allocation technique (*see* Finkelstein, [0034]) or Felter’s modified pro rata power allocation technique (*see* Felter, [0061]), what the power management logic does with those allocations remains the same: it adjusts the frequency (and possibly also voltage) of the associated power domains according to the power allocations.

176. It is therefore my opinion that Finkelstein’s power management logic, as adapted to perform Felter’s modified pro rata technique, controls a frequency of

the first processor core power plane (first domain) and a frequency of the second processor core power plane (second domain) based on the allocated portions of the power/energy budget.

4. **Claim 12: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”**

177. The combination of Finkelstein and Felter at least renders obvious the additional features of claim 12 because, as modified, Finkelstein’s power management logic would “throttle[e]” (not grant) increased processor core activity when the processor core attempts to perform activity that would “exceed” its budget-related “limit” during a time interval. *See* Felter, [0065]; *see also* Section IX.A.2 above (discussing Ground 1, claim 12). In response to such a request for higher frequency being denied (by denying the processor core the ability to perform more instructions during the allotted time period), a POSITA would have at least found it obvious to increase that processor core’s frequency during a subsequent interval by allocating more power to it. *See* Felter, [0051] (“In perhaps the simplest implementation, the predicted activity for time period N+1 is the measured activity for time period N. This approach relies on the not unreasonable assumption that very recent activity is a good predictor of current activity. This approach is also desirable

for its responsiveness in the context of a rapidly changing operating environment.”), [0066] (“When the current interval expires, power administrator 200 generates new predictions for the next upcoming period in block 304 and the process repeats itself indefinitely.”).

178. As I explain above for feature [8c], the combination of Finkelstein and Felter would allocate power to first and second power planes (domains) in a way that accounts for the “standby power requirements [$E_{\min}^{k,i}$] (which must always be met) and a portion of the available power [$E^k - \text{SUM}(E_{\min}^{k,i})$].” Felter, [0057]-[0061]; *see also* Section IX.C.3.d above (describing minimum power reservations in the combination of Finkelstein and Felter in the context of feature [8c]). This “represents a technique for distributing scarce system power equitably so that the system performance of each component is impacted proportionately” and “in proportion to the requirement for the components by the workload(s) executing on the system.” Felter, [0061]. According to Felter’s modified pro rata technique, Finkelstein’s power management logic, as modified by Felter, would then “deriv[e] . . . activity limits that correspond to the power budgets allocated.” Felter, [0064]; *see also* Felter, FIG. 3 (block 308).

179. Felter further discloses that if either of the components “exceed” their corresponding limit during a given interval—that is, it exceeds its allocated power budget—then the power administrator throttles that component’s activity “until the

next monitoring interval begins.” Felter, [0065]. Accordingly, a POSITA would have understood that if any of Finkelstein’s power planes “exceed” their corresponding activity limit in the Finkelstein-Felter combination, the modified power management logic would throttle activity “until the next monitoring interval begins.” Felter, [0064]-[0066]; *see also* Felter, FIG. 3. It would then “generate[] new predictions for the next upcoming period” and allocate power to the throttled components according to their new needs. Felter, [0066]; *see also* Finkelstein, [0039]-[0040] (explaining that, after one time period expires, “the time step may be incremented (e.g., by logic 140), e.g., to determine future values for each power plane budget during a next time period” and that this “may be repeated continuously (or on a periodic basis)”).

180. To that end, Felter explains that “the predicted activity for time period N+1 is a simple function of the monitored activity for time period N.” Felter, [0051]. In one example, Felter explains the following:

[T]he predicted activity for time period N+1 is the measured activity for time period N. This approach relies on the not unreasonable assumption that very recent activity is a good predictor of current activity. This approach is also desirable for its responsiveness in the context of a rapidly changing operating environment.

Felter, [0051]. Likewise, Finkelstein discloses an “energy budget defined iteratively as follows: $E_{n+1} = \alpha E_n + (TDP_n - P_n) \Delta t_n$.” Finkelstein, [0028]-[0029]. As this equation

makes clear, the budget for time period $N+1$ (E_{n+1}) is a simple function of the monitored budget for time period N (E_n), just as Felter discloses. This approach facilitates “responsiveness in the context of a rapidly changing operating environment.” Felter, [0051]. Since Felter’s modified pro rata technique—which the Finkelstein-Felter combination employs—uses the amount of recent activity to allocate power over the next interval, a POSITA would have understood that if one of Finkelstein’s power planes (domains) was throttled during the previous time interval, then its $W_{k,i}$ value should be incremented for the next interval. *See* Section IX.A.2 above (explaining that if one of Felter’s components was throttled during the previous time interval, then its predicted utilization ($UTIL_N$) should be increased for the next interval), Section IX.C.3.e above (explaining that Finkelstein’s $W_{k,i}$ values in the Finkelstein-Felter combination are analogous to Felter’s $PACT_N/P_{TOT}$ values, which are proportional to $UTIL_N$). In other words, if one of Finkelstein’s power planes (domains) was prevented from performing more tasks during the previous time interval (denying its request to perform at a higher frequency), then a POSITA would have understood from Felter’s disclosures that the power plane (domain) should be granted a higher frequency during the next budgeted time interval.

181. It was well known in the art to increase the frequency of a computing domain when its power needs are not satisfied. For example, Therien describes a “policy manager” that “periodically evaluates” a system “to determine whether any

power budget adjustments are appropriate” and “make[s] budget adjustments as conditions associated with the system 400 change (e.g., when a component raises or lowers an increase request flag).” Therien, [0031]. Similarly, Conroy explains that “the power of the system may be shifted toward the subsystem having a higher workload” such that “higher power draw is allocated to and allowed for the subsystem having a higher workload.” Conroy, [0006]. This is also common sense. A component that is getting busier than it was before should be given additional resources to work faster if it is unable to keep pace with system demands.

182. Thus, in the combination of Finkelstein and Felter, it is my opinion that a POSITA would have found it obvious to increment $W_{k,1}$ (the first sharing policy value) when a request for a higher frequency on the first domain (first processor core power plane, e.g., $i=1$) is not granted (throttled), and to increment $W_{k,2}$ (the second sharing policy value) when a request for a higher frequency on the second domain (second processor core power plane, e.g., $i=2$) is not granted (throttled).

5. Independent Claim 13

a. [13pre]: “A system comprising:”

183. The combination of Finkelstein and Felter includes feature [13pre] because Finkelstein discloses a “computing system” with, among other things, “one or more processors,” a “power source,” “voltage regulator(s),” and “power management logic.” Finkelstein, [0010]-[0014]; *see also* Finkelstein, [0042]-[0047],

FIGS. 1, 6. Finkelstein's Figure 1, reproduced below, illustrates an exemplary computing system 100.

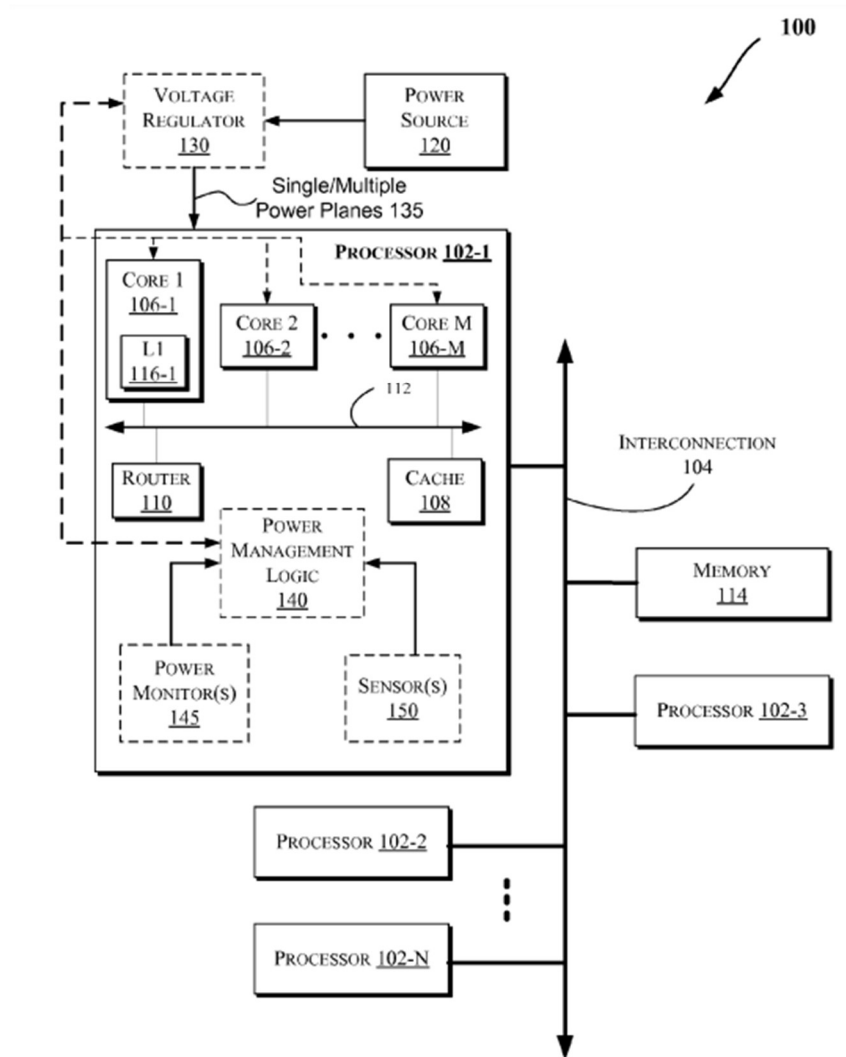


FIG. 1

Finkelstein, FIG. 1

184. In the combination of Finkelstein and Felter I have considered and described in Sections IX.C.1-2, a POSITA would have found it obvious to adapt Finkelstein's computing system such that its power management logic performs

power allocation using Felter's modified pro rata technique. Finkelstein's computing system, as modified in this way, constitutes a "system" as recited in feature [13pre].

b. [13a]: "a multicore processor having a first domain including a plurality of cores, a second domain including a graphics engine, and a third domain including system agent circuitry,"

185. The combination of Finkelstein and Felter includes feature [13a] for essentially the same reasons I discuss above in the context of Ground 2. Finkelstein's computing system exemplary computing system comprises a multicore, "multiple power plane" processor 102-1 (multicore processor) that includes, among other things, the following power domains: (1) a first power plane (domain) to power at least first and second processor cores 106-1 and 106-2, (2) a second power plane (domain) to power a "graphics . . . controller[]" (graphics engine) that facilitates "Graphics Effect(s) (GFX)," and (3) a third power plane (domain) to power Finkelstein's "power management logic" (system agent circuitry). Finkelstein, [0011], [0014], [0026], [0046], FIGS. 1, 6. In the combination of Finkelstein and Felter, Finkelstein's power management logic applies Felter's modified pro rata power allocation technique, but this does not affect the definition of these three domains.

c. [13b]: “the third domain to operate at a fixed power budget and to dynamically allocate a variable power budget between the first and second domains, wherein”

186. The combination of Finkelstein and Felter includes feature [13b], as I explain below.

187. First, Finkelstein’s third domain (the power management logic, adapted to perform Felter’s modified pro rata power allocation technique) dynamically allocates a variable power/energy budget (denoted E^k)⁶ between a group of processor

⁶ Although Finkelstein allocates an energy budget and Felter allocates a power budget, a POSITA would have understood the two to be equivalent. A POSITA would have understood a power budget can be converted to an energy budget, and vice versa, simply by multiplying the power budget by the time interval (or dividing the energy budget by the time interval). *See* Finkelstein, [0028] (“ $E_{n+1} = \alpha E_n + (TDP_n - P_n)\Delta t_n$ ”). The ’316 patent itself also makes clear that the claimed “power budget” may be expressed in terms of energy (using Joules), not just in terms of power (Watts). *See* ’316 patent, 3:22-62 (describing equation [1]), claim 7 (defining “the power budget for the current time interval” as an energy “ E_n ” and saying “to determine the power budget” in terms of “energy”). A POSITA thus would have

cores (first domain) and a graphics controller (second domain) that performs a “GFX intensive workload.” *See* Finkelstein, [0009] (“[I]n designs with multiple power planes, power may be distributed among different power planes under energy-based definitions”), [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”), [0028] (“Energy-based power management may be performed by controlling the energy budget defined iteratively as follows: $E_{n+1} = \alpha E_n - (TDP_n - P_n) \Delta t_n$ (1)”), [0034] (“In the general case, power plane i may obtain the portion of the budget: $E^{k,i} = W_{ki} E^k$ (2)”). The allocated power budget is variable because it is determined “iteratively” for each new time interval based on the power/energy not consumed during the previous one. Finkelstein, [0028]; *see also* Finkelstein, [0039]-[0040] (explaining that, after one time period expires, “the time step may be incremented (e.g., by logic 140), e.g., to determine future values for each power plane budget during a next time period” and that this “may be repeated continuously (or on a periodic basis)”), FIG. 5 (showing the budget is determined for each time step).

understood the claimed “power budget” refers interchangeably to power (in Watts) and energy (in Joules).

188. Finkelstein's power management logic, as adapted to perform Felter's modified pro rata power allocation technique, receives information about individual and global constraints, and it uses those constraints (denoted by the index k) to determine a power/energy budget (E^k) per constraint (k). Finkelstein, [0039] (“[A]t an operation 502, the energy budget remainder per constraint E_n^k may be determined (e.g., by the logic 140 in accordance with formula (1) . . .).”). That budget may be “a common package power/energy budget” shared among the processor and graphics power planes. Finkelstein, [0026]; *see also* Finkelstein, [0034] (“In the general case, power plane i may obtain the portion of the budget: $E^{k,i} = W_{ki} E^k$ (2)”). As I explain above in the context of claim 8, Finkelstein's power management logic, as modified in the combination, dynamically allocates the power/energy budget between the first domain (e.g., $i=1$) and the second domain (e.g., $i=2$) by applying the following equation:

$$E^{k,i} = E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i}) \quad (2)$$

See Felter, [0061], FIG. 5 (blocks 504, 508); Finkelstein, [0034]; Sections IX.C.1-2 above (describing the combination of Finkelstein and Felter). In this manner, the power management logic (third domain) dynamically allocates a variable power budget (E^k) between the processor core power plane (first domain) and the graphics controller power plane (second domain).

189. Second, as I explained for [13b] in the context of Ground 2, a POSITA would have understood that Finkelstein's power management logic operates using a fixed power budget. *See* Section IX.B.1.c above (explaining why a POSITA at least would have found it obvious to operate Finkelstein's power management logic using a fixed power budget and discussing Finkelstein, [0016], [0024], [0031], [0039]; Conroy, [0081], [0094], [0146], [0165], [0203]). In the combination of Finkelstein and Felter, Finkelstein's power management logic applies Felter's modified pro rata power allocation technique, but this does not affect how power is supplied to the power management logic itself (third domain).

d. [13c]: “the system agent circuitry includes a power sharing logic to determine the variable power budget for a current time interval and”

190. The combination of Finkelstein and Felter includes feature [13c] for essentially the same reasons I explain regarding feature [8a], which recites, in part, the following: “determining, in a power controller of a multi-domain processor, a power budget for the multi-domain processor for a current time interval.” *See* Section IX.C.3.b above (discussing feature [8a] in the context of Ground 3). The only difference regarding how the Finkelstein-Felter combination applies to feature [13c] (as compared to feature [8a]) is that the variable power budget for feature [13c] applies specifically to a processor core power plane with multiple cores (first domain) and a graphics controller power plane (second domain). *See* Finkelstein,

[0046] (explaining that the processor core power plane and graphics controller power plane may “share a common package power/energy budget”). The variable power budget for feature [8a], on the other hand, applies to any two power planes that share a power/energy budget, such as two processor core power planes (first and second domains). *See* Section IX.C.3.b above (discussing feature [8a] in the context of Ground 3).

191. It is therefore my opinion that the combination of Finkelstein and Felter includes a power sharing logic (namely, Finkelstein’s power management logic, as adapted to perform Felter’s modified pro rata power allocation technique) for determining the variable power budget (E_{n+1}^k) for a current time interval ($n+1$). *See* Finkelstein, [0028] (“Energy-based power management may be performed by controlling the energy budget defined iteratively as follows: $E_{n+1} = \alpha E_n - (TDP_n - P_n) \Delta t_n (1) \dots$ ”), [0039]-[0040] (explaining that, after one time period expires, “the time step may be incremented (e.g., by logic 140), e.g., to determine future values for each power plane budget during a next time period” and that this “may be repeated continuously (or on a periodic basis)”).

- e. **[13d]: “to allocate a first portion of the variable power budget to the first domain according to a first power sharing value for the first domain, and to allocate a second portion of the variable power budget to the second domain according to a second power sharing value for the second domain; and”**

192. For the same reasons I explain in the context of features [8c] and [8d] above, the combination of Finkelstein and Felter includes feature [13d]. In the combination of Finkelstein and Felter, where Finkelstein’s power management logic applies Felter’s modified pro rata technique, Finkelstein’s power allocation equation (2) would be modified as follows:

$$E^{k,i} = E_{min}^{k,i} + W_{k,i}(E^k - \sum_i E_{min}^{k,i}) \quad (\text{Eq. 2, modified})$$

where $E_{min}^{k,i}$ corresponds to Felter’s minimum power / standby power values $PMIN_N$, the term $(E^k - \sum_i E_{min}^{k,i})$ corresponds to Felter’s P_{AVAIL} (the remainder of the power budget after allocating the minimum power values to each power plane), and $W_{k,i}$ corresponds to Felter’s $PACT_N/PACT_{TOT}$ values (the relative proportions of the remaining power budget allocated to each power plane). *See* Section IX.C.1-2 above (describing the combination of Finkelstein and Felter). It is therefore my opinion that, in the combination of Finkelstein and Felter, Finkelstein’s power management logic, as modified, would allocate a first portion ($E^{k,1}$) of the variable power budget (E^k) to the first domain (processor core power plane) according to a first power sharing value ($W_{k,1}$) for the first domain, and that it would also allocate a

second portion ($E^{k,2}$) of the variable power budget (E^k) to the second domain (graphics controller power plane) according to a second power sharing value ($W_{k,2}$) for the second domain.

f. [13e]: “a dynamic random access memory (DRAM) coupled to the multicore processor.”

193. For essentially the same reasons I explain above in the context of feature [13e] of Ground 2, the combination of Finkelstein and Felter includes feature [13e]. Finkelstein’s exemplary computing system includes “memory” that “may communicate with the processors . . . via [an] interconnection.” Finkelstein, [0013]. It “may store data, including sequences of instructions that are executed by the processor” and may comprise “dynamic RAM (DRAM).” Finkelstein, [0043]. In the combination of Finkelstein and Felter, Finkelstein’s power management logic applies Felter’s modified pro rata power allocation technique, but this does not affect the fact that Finkelstein’s computing system includes a DRAM coupled to exemplary multicore processor 102-1.

6. Claim 14: “The system of claim 13, wherein the power sharing logic is to dynamically allocate substantially all of the variable power budget to the first domain for a first workload, and to dynamically allocate substantially all of the variable power budget to the second domain for a second workload executed after the first workload.”

194. The combination of Finkelstein and Felter includes the additional features of claim 14 for essentially the same reasons I provide in the context of

claim 14 for Ground 2. *See* Section IX.B.2 above (describing how Finkelstein satisfies the additional features of claim 14). Finkelstein's power management logic, as adapted to perform Felter's modified pro rata power allocation technique, distributes the power/energy budget in a "temporal" way that accounts for the workload during each interval. Finkelstein, [0027]-[0029]. Moreover, the power management logic allocates the "entire budget" to a "single power plane" when needed. Finkelstein, [0034]; *see also* Finkelstein, [0026] ("[I]t is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.").

195. Finkelstein's power management logic, as modified, may repeat its power management functions "continuously (or on a periodic basis) . . . for multiple cores sharing the same power plane." Finkelstein, [0025]; *see also* Finkelstein, [0040] (similar); Felter, [0066] ("When the current interval expires, power administrator 200 generates new predictions for the next upcoming period in block 304 and the process repeats itself indefinitely."). For example, "the time step may be incremented (e.g., by logic 140), e.g., to determine future values for each power plane budget during a next time period." Finkelstein, [0039]; *see also* Finkelstein, FIG. 5; Felter, [0066]. Finkelstein's power management logic, as modified, does so by dynamically allocating a variable power budget (E^k) between the processor core

power plane (first domain) and graphics controller power plane (second domain) using Felter's modified pro rata technique. *See* Sections IX.C.1-2 above (describing the combination of Finkelstein and Felter), Section IX.C.5.e (describing how the power/energy budget is allocated in the Finkelstein-Felter combination). It is therefore my opinion that a POSITA would have understood Finkelstein's power management logic, as modified according to Felter, uses the workload performed during a first time period to dynamically allocate a variable power budget to the first and second domains. Likewise, it is my opinion that a POSITA would have understood that Finkelstein's power management logic, as modified according to Felter, subsequently uses the workload performed during a second time period to dynamically re-allocate a variable power budget to the first and second domains. *See* Felter, [0054] ("[P]ower administrator 200 includes allocating . . . power budgets to the various system components based on the predicted level of activity."), [0051] ("[T]he predicted activity for time period N+1 is a simple function of the monitored activity for time period N (or for another recent time period)."), [0066] ("When the current interval expires, power administrator 200 generates new predictions for the next upcoming period in block 304 and the process repeats itself indefinitely."). In this way, Finkelstein's power management logic, as modified according to Felter, determines a variable power budget "for multiple cores sharing the same power

plane” and allocates a portion of it among those power planes “continuously (or on a periodic basis).” Finkelstein, [0025]; *see also* Finkelstein, [0040]; Felter, [0066].

196. The combined teachings of Finkelstein and Felter at least suggest that substantially all of the variable power/energy budget would be allocated to the first domain (processor core power plane) under certain conditions. For example, a first workload during a first time interval may require the processor cores to perform substantially all of the necessary tasks because the graphics controller would be unable or inefficient in doing so. Then, during a subsequent time interval, a second workload may require the graphics controller to perform substantially all of the necessary tasks because the processor cores would be unable or inefficient in doing so. For example, Finkelstein teaches to share a “common package power/energy budget” between the processing and graphics power planes and explains that “power management may need to be satisfied . . . per package.” Finkelstein, [0026]. According to Finkelstein, “the entire” shared package power budget also may be allocated to one of the power planes during a given time interval. For example, “a *single power plane* may obtain *the entire budget*” based on “user (or application defined) preferences that describe how budget E^k is distributed among power planes.” Finkelstein, [0034]. Finkelstein thus at least suggests that the power management logic, as modified according to Felter, would allocate substantially all of the variable power budget to one power plane (e.g., the processor cores) when the

“user (or application defined) preferences” (i.e., the $W_{k,i}$ values) for the first workload make it appropriate to do so. *See, e.g.*, Felter, [0054] (explaining that Felter’s power administrator 200 (analogous to Finkelstein’s power management logic 140) “allocate[es] . . . power budgets to the various system components based on the predicted level of activity”); Section IX.C.2 (describing how the $W_{k,i}$ values in the Finkelstein-Felter combination depend on utilization).

197. A POSITA would have understood that the expected activity for the processor cores would be high for a processor-heavy workload, whereas the expected activity for the graphics controller would be low. *See* Felter, [0059]-[0061] (explaining that power is allocated “in proportion to the requirement for the components by the workload(s) executing on the system”). Finkelstein explains, for example, that “it is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.” Finkelstein [0026]. Based on these disclosures, for example, a POSITA would have understood that the processor cores (first domain) do not require very much power when the system is given a graphics-heavy workload. Substantially all of the power budget would be allocated to the graphics controller (second domain), which only makes sense for a graphics-heavy workload. In other words, a POSITA would have understood that the power sharing value for

the first domain ($W_{k,1}$) would be near 0 for a graphics-heavy workload, whereas the power sharing value for the second domain ($W_{k,2}$) would be near 1. Thus, substantially all the power budget would be allocated to the graphics power plane (second domain). For a processor-heavy workload, the situation would be reversed.

198. This type of alternation between computation-heavy and graphics-heavy workloads was well-known in the art, and it is common sense. *See* Gwennap, 6 (“If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs.”); Therien, [0017] (“[A] computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU.”); Bose, Abstract (“During the first mode, instructions associated with applications are executed by a first asymmetric core, and a second asymmetric core is inactive. During the second mode, the instructions are executed by the second asymmetric core, and the first asymmetric core is inactive.”), [0010] (disclosing “reassigning threads of the workload to cores remaining active by assigning threads based on priority constraints and thread execution history to improve the operational efficiency of the multi-core architecture”), [0021] (“[C]ore activation/ deactivation may be provided

based upon hardware or priority constraints.”), [0085] (describing a “thread scheduler” that “priorit[izes]” and “assigns” power distribution to “high performance threads” over “low performance” threads).

199. For example, a POSITA would have understood that many applications, like program development tools and scientific applications, produce a heavy load for the processors and almost no load for the graphics controller. *See e.g.*, Conroy, [0196] (“For example, program development tools and scientific applications present a high load to the CPU, but almost no load to the GPU that leads to an asymmetric load profile of the system (e.g. the CPU consumes a lot more power than the GPU).”). Other applications, such as advanced user interfaces or graphics editing applications, present the opposite situation. Conroy, [0196] (“Advanced user interfaces or graphics editing application present a high load to the GPU and a modest load to the CPU that leads to another asymmetric load profile to the system.”). Accordingly, “[m]any professional applications present an alternating high workload to the CPU and to the GPU.” Conroy, [0196]. Again, it is only common sense that the processor cores would receive substantially all of the power when the workload is devoid of graphics, and the graphics controller would receive substantially all of the power when the workload is graphics-heavy.

200. Accordingly, it is my opinion that a POSITA would have expected a multi-domain processing system (like the one in the Finkelstein-Felter combination)

to alternate between executing processor-heavy applications and graphics-heavy applications. Based on the disclosures in Finkelstein and Felter and common knowledge about workloads in a typical computing system, a POSITA would have understood that Finkelstein's power management logic, as modified according to Felter, would allocate substantially all of the power budget E^k to the processor core power plane (first domain) for a processor-heavy workload and substantially all of the power budget to the graphics controller (second domain) for a graphics-heavy workload. *See* Finkelstein [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”), [0034] (“[A] single power plane may obtain the entire budget, so the corresponding weight vector is a unit vector.”); Felter, [0059]-[0061] (explaining that power is allocated “in proportion to the requirement for the components by the workload(s) executing on the system”).

201. Thus, in my opinion, to efficiently allocate power to the processor core power plane (first domain) for processor-heavy workloads and to efficiently allocate power to the graphics controller power plane (second domain) for graphics-heavy workloads, a POSITA would have been motivated to allocate substantially all of the variable power budget to the graphics controller (second domain) for a graphics-heavy second workload submitted after a processor-heavy first workload. *See*

Gwennap, 6 (“If the GPU [(Graphics Processing Unit)] is not in use or is operating at a low speed, the power manager can assign some of its thermal budget to the CPUs [Central Processing Units], allowing them to run at a higher clock speed. Conversely, on a graphics-intensive application, the power manager can overclock the GPU while slowing down the CPUs.”); Conroy [0196] (“Various applications provide various workloads to each of the subsystems. For example, program development tools and scientific applications present a high load to the CPU, but almost no load to the GPU that leads to an asymmetric load profile of the system (e.g. the CPU consumes a lot more power than the GPU). Many professional applications present an alternating high workload to the CPU and to the GPU that results in an alternating asymmetric load profile of the system. Advanced user interfaces or graphics editing application present a high load to the GPU and a modest load to the CPU that leads to another asymmetric load profile to the system.”).

202. A POSITA would have reasonably expected to succeed in configuring Finkelstein’s power management logic, as modified according to Felter, to allocate power to alternating workloads in this way because it would have required no more than routine skill in the art. For example, because power distribution preferences may be “user (or application defined)” (Finkelstein, [0034]), a POSITA would have known to program the power management logic to allocate substantially all of the

available power to the processor core power plane (first domain) for first workload when it contains a processor-heavy workload with little or no graphics operations to be performed. Likewise, a POSITA would have known to program the power management logic to allocate substantially all of the available power to the graphics controller (second domain) when the workload during a subsequent interval is graphics-heavy and requires little or no support from the processor cores. *See* Finkelstein, [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”). Doing so would have involved known techniques, e.g., writing software of the kind those ordinarily skilled in the art would have been able to write. *See, e.g.,* Therien, [0014] (“[A]ny of the methods described herein may be performed by hardware, software (including lower level code, such as microcode), or a combination of hardware and software.”), Conroy, [0191] (“[T]he techniques may be carried out in a computer system or other data processing system in response to its processor, such as a microprocessor or a microcontroller, executing sequences of instructions contained in a memory”), [0195] (“The power used by at least a subset of each of the subsystems is controlled, e.g., by a microcontroller”), [0196] (“[T]he load profile may be identified using workloads determined by measuring/sensing power (e.g. current drawn) by each

subsystem or by measuring power for certain subsystems and estimating or predicting power for other subsystems or by estimating power for all subsystems. In another embodiment, the load profile may be identified using workloads determined by the operating system out from historical scheduling data. In yet another embodiment, to identify the load profile of the system, the information about the workload of the subsystem provided by an application may be used.”).

203. Accordingly, it is my opinion that a POSITA would have found it obvious in the Finkelstein-Felter combination to dynamically allocate substantially all of the variable power budget (E_n^k) to the processor core power plane (first domain) for a first (processor-heavy) workload, and to dynamically allocate substantially all of Finkelstein’s variable power budget (E_{n+1}^k) to the graphics controller power plane (second domain) for a second (graphics-heavy) workload executed after the first (processor-heavy) workload.

7. **Claim 15: “The system of claim 13, wherein the power sharing logic is to increment the first power sharing value when a request for a higher frequency for the first domain is not granted, and to increment the second power sharing value when a request for a higher frequency for the second domain is not granted.”**

204. The combination of Finkelstein and Felter includes the additional features of claim 15 for essentially the same reasons I explain above regarding claim 12, which recites the same additional claim features. To summarize briefly, a

POSITA would have found it obvious to calculate the power sharing values ($W_{k,i}$) according to the expected activity (UTIL) of the corresponding power plane i and thus to increment those values when the expected activity of the corresponding power plane i increases (to meet a request for a higher frequency). *See* Section IX.C.4 above (explaining why the additional features of claim 12 would have been obvious based on the Finkelstein-Felter combination).

8. **Claim 16: “The system of claim 13, wherein the power sharing logic is to further allocate the first portion of the variable power budget according to a first minimum reservation value for the first domain and allocate the second portion of the variable power budget according to a second minimum reservation value for the second domain.”**

205. The combination of Finkelstein and Felter includes the additional features of claim 16 for essentially the same reasons I explain regarding feature [8c], which recites the following: “allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain.” *See* Section IX.C.3.d above (discussing feature [8c] in the context of Ground 3). The only difference regarding how the Finkelstein-Felter combination applies to claim 16 (as compared to feature [8c]) is that the “second domain” for claim 16 is a graphics controller power plane rather than a second processor core power plane. *See* Finkelstein, [0046] (explaining that the processor core power plane and graphics controller power plane may “share a common package power/energy budget”). The

second domain in the context of feature [8c], on the other hand, can be any second processor core power plane. *See* Section IX.C.3.d above (discussing feature [8c] in the context of Ground 3). As I explain above in the context of features [8c] and [8d], Finkelstein's power management logic, as adapted to perform Felter's modified pro rata power allocation technique, allocates a first portion ($E_{k,1}$) of the variable power budget (E^k) to the first domain (first power plane) according to a first sharing policy value / power sharing value ($W_{k,1}$) and a first minimum reservation value ($E_{\min}^{k,1}$), and it allocates a second portion ($E_{k,2}$) of the power budget (E^k) to the second domain (second power plane) according to a second sharing policy value / power sharing value ($W_{k,2}$) and a second minimum reservation value ($E_{\min}^{k,2}$). *See* Sections IX.C.3.d-e above (describing how Finkelstein's power management logic, as modified according to Felter, allocates power); Finkelstein, [0026], [0044]-[0046]; Felter, [0061], FIG. 5.

9. Claim 17: "The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain."

206. The combination of Finkelstein and Felter includes the additional features of claim 17 for essentially the same reasons I provide in the context of claim 17 for Ground 2. *See* Section IX.B.3 above (describing how Finkelstein satisfies the additional features of claim 17). Finkelstein's power management logic,

as modified according to Felter, allocates the “entire budget” to a “single power plane” when needed, based on “user (or application defined) *preferences*.” Finkelstein, [0034] (emphasis added). A POSITA would have understood this to be true when the processor cores (first domain) and graphics controller (second domain) share a common package budget. For example, Finkelstein explains that “it is possible to share a common package power/energy budget and allow for use of for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.” Finkelstein, [0026].

207. For the same reasons I explain in the context of claim 17 for Ground 2, it is my opinion that a POSITA would have found it obvious in the Finkelstein-Felter combination to allocate the available variable power budget ($E^k - \sum_i E_{min}^{k,i}$) to the graphics controller (second domain) according to a preference value that favors the graphics controller (second domain) over the processor cores (first domain) (e.g., by setting $W_{k,2} > W_{k,1}$, such that $W_{k,2}$ would represent a preference value). *See* Section IX.B.3 above (describing how Finkelstein satisfies the additional features of claim 17), Sections IX.C.3.d-e (describing how the power management logic allocates power in the combination of Finkelstein and Felter); Finkelstein, [0026], [0034]; Felter, [0051], [0059]-[0061], FIG. 5.

D. Ground 4: The Finkelstein-Felter-Therien Combination Renders Obvious Claims 12, 15, and 17

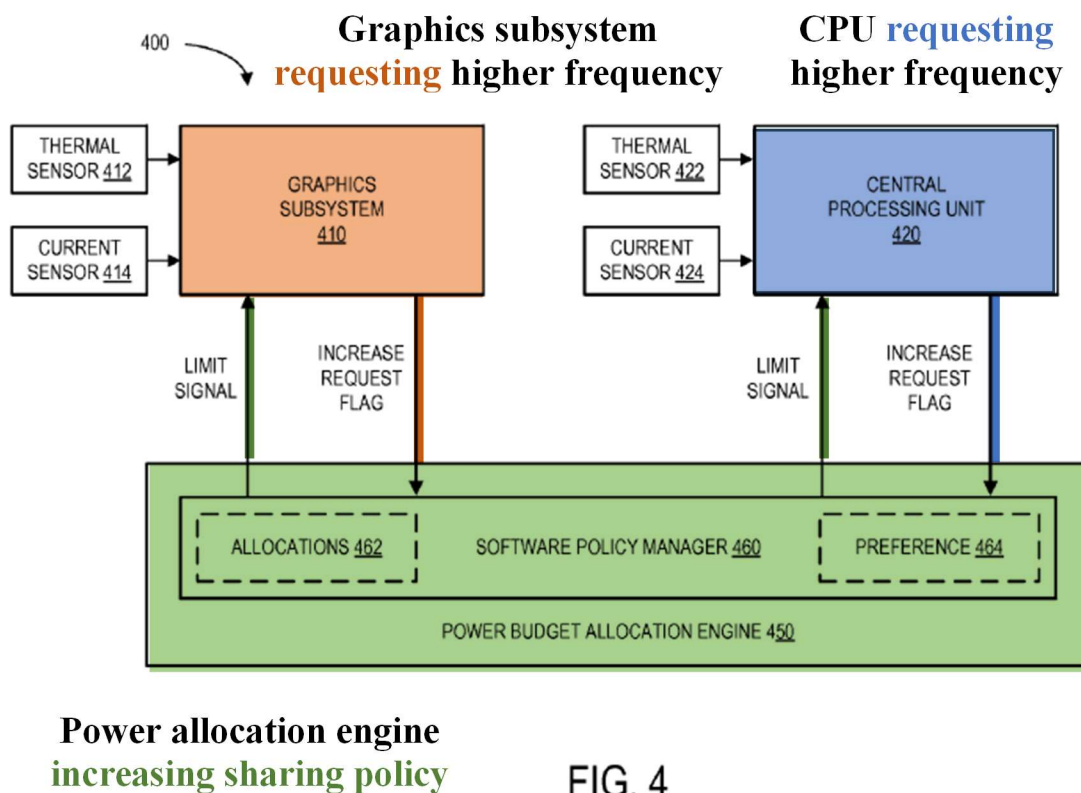
1. **Claim 15: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”**

208. It is my opinion that the combined teachings of Finkelstein and Felter, further combined with the teachings of Therien, renders claim 15 obvious. A POSITA would have found it obvious from Therien’s disclosures to program Finkelstein’s power management logic, as modified according to Felter, to increment the policy sharing value ($W_{k,i}$) when corresponding power plane i requests a higher frequency and has that request denied.

209. Therien describes similar computing systems to those in Finkelstein and Felter. Moreover, just like Finkelstein, Therien explains that its power management techniques can be used, for example, with a “graphics subsystem” and “CPU” (processor) that share a power budget. *See* Therien, [0015] (“[A] power budget allocation engine of a multicomponent computer system may receive a power budget allocation adjustment request signal from a first component. For example, the power budget allocation adjustment request signal might be an ‘increase request flag’ that can be raised (or lowered) by the first component. Note that the first component might be associated with, by way of examples only, a processor such as

a CPU or a graphics subsystem such as a Graphics and Memory Controller Hub ('GMCH')."), [0025]-[0026] ("[T]he computing system 400 might have an overall power and/or thermal budget that can be allocated among the graphics subsystem 410 and CPU 420."), FIGS. 4-5. Therien explains that adjusting the power budget allocation for a component may be appropriate if "an additional allocation of power could be usefully consumed." Therien, [0019]. For example, "the graphics subsystem might evaluate a current frame rate, render p-state, render frequency, and/or application being executed by the graphics subsystem (e.g., a graphics-intensive 3D game) before requesting an increase in its power allocation. Thereien, [0020]. Or a CPU "might evaluate an instruction execution rate and/or an application being executed by the CPU before requesting an increase in its power allocation." Therien, [0020]. If additional power allocation would be useful, Therien explains that "an increase request flag can be raised (or lowered)," and the system may experience an "increase[d] . . . frame rate" or an "increase[e] [in] the number of instructions retired per second." Therien, [0021]-[0022]. A POSITA would have understood that such results are achieved by increasing the component's operating frequency. For example, Therien's power budget allocation engine may "adjust a power budget allocation signal provided to the . . . component" so that "an individual *power limit imposed on a component* . . . may be relaxed or *raised* as appropriate." Therien, [0013]-[0016] (emphasis added). Therien's Figure 4, which I have

reproduced below with annotations, illustrates Therien's process. *See* Therien, [0011] (“[T]he first component 110 might transmit a power budget allocation adjustment request signal to the power budget allocation engine 150. . . . Similarly, the second component 120 might (or might not) determine that it would be able to utilize additional power and transmit a power budget allocation adjustment request signal to the power budget allocation engine 150.”), [0014]-[0023] (describing Therien's Figure 2), [0024]-[0031] (describing Therien's Figure 4), FIGS. 2, 4.



Therien, FIG. 4 (annotated)

210. As I explain above in the context of claims 12 and 15 for Ground 3, Finkelstein's power management logic, as modified according to Felter, dynamically allocates power to various power planes based on their anticipated activity. *See* Sections IX.C.4 and IX.C.7 above (describing how the Finkelstein-Felter combination satisfies claims 12 and 15). Felter teaches to use recent activity levels to predict a component's upcoming activity levels. *See* Felter, [0051]-[0054] ("[P]ower administrator 200 includes allocating . . . power budgets to the various system components based on the predicted level of activity."). A POSITA would have understood that, if the allocated power during the previous time interval was not sufficient, then "an additional allocation of power could be usefully consumed." Therien, [0019]. Accordingly, a POSITA would have understood from Therien's disclosures that the component should request additional power for the next time interval. *See* Therien, [0019]-[0022] (describing Therien's Figure 3 and explaining that "an increase request flag that can be raised (or lowered) . . . based on a thermal condition detected by the first component and/or a power consumption condition"), [0024]-[0031] (describing Therien's Figure 4), FIGS. 3-4. In the context of the Finkelstein-Felter combination, a POSITA thus would have been motivated to respond by incrementing Felter's predicted activity (UTIL) for the upcoming interval, which would increment the power sharing value ($W_{k,i}$) as a result. *See* Section IX.C.2 above (describing the relationship between a component's predicted

activity (UTIL) and its power sharing value $W_{k,i}$ in the Finkelstein-Felter combination); Felter, [0051], [0059]-[0061], [0066], FIG. 5.

211. A POSITA would have reasonably expected to succeed in doing this because it would have required nothing beyond routine and ordinary skill in the art. As I explain in the context of Ground 3, Finkelstein's power management logic, as modified according to Felter, already adjusts the sharing policy values / power sharing values ($W_{k,i}$) based on the expected activities (UTIL) of the associated power planes. *See See* Section IX.C.2 above (describing the relationship between a component's predicted activity (UTIL) and its power sharing value $W_{k,i}$ in the Finkelstein-Felter combination); Felter [0051]-[0054] (“[P]ower administrator 200 includes allocating . . . power budgets to the various system components based on the predicted level of activity.”), [0059]-[0061] (describing Felter's modified pro rata power allocation technique). Incrementing the sharing values ($W_{k,i}$) to allocate additional power in response to a request for a higher frequency (as Therien discloses) would not have required anything beyond well-known techniques (e.g., writing “lower level code, such as microcode”) in accordance with their known use (power budgeting and management). *See* Therien, [0014] (“Note that any of the methods described herein may be performed by hardware, software (including lower level code, such as microcode), or a combination of hardware and software.”); *see* Finkelstein, [0039] (describing the basic operations of Finkelstein's power

management logic); Felter [0051]-[0066] (describing how to determine and allocate a power budget based on activity levels), FIGS. 3, 5.

212. It is therefore my opinion that a POSITA would have found it obvious in view of Therien's disclosures to further configure Finkelstein's power management logic, as modified according to Felter, to increment the first sharing policy value ($W_{k,1}$) when a request for a higher frequency for the first domain (processor core power plane) is not granted, and to increment the second sharing policy value ($W_{k,2}$) when a request for a higher frequency for the second domain (graphics controller power plane) is not granted.

2. Claim 12: "The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted."

213. The combination of Finkelstein, Felter, and Therien includes the additional features of claim 12 for essentially the same reasons I explain above regarding claim 15, which recites the same additional claim features. That is, a POSITA would have found it obvious in view of Therien's disclosures to further configure Finkelstein's power management logic, as modified according to Felter, to increment the first sharing policy value ($W_{k,1}$) when a request for a higher frequency for the first domain (a first processor core power plane) is not granted,

and to increment the second sharing policy value ($W_{k,2}$) when a request for a higher frequency for the second domain (a second processor core power plane) is not granted. *See* Section IX.D.1 above (describing how the combination of Finkelstein, Felter, and Therien satisfies claim 15 for Ground 4); Finkelstein, [0034], [0039]; Felter, [0051]-[0054], [0059]-[0061], [0066]; Therien, [0019]-[0022], FIGS. 1-3. The only difference regarding how the Finkelstein-Felter-Therien combination applies to claim 12 (as compared to claim 15) is that the “second domain” for claim 15 is a graphics controller power plane rather than a second processor core power plane. *See* Finkelstein, [0046] (explaining that the processor core power plane and graphics controller power plane may “share a common package power/energy budget”). The second domain in the context of claim 12, on the other hand, can be any second processor core power plane. *See* Section IX.D.1 above (describing how the combination of Finkelstein, Felter, and Therien satisfies claim 15 for Ground 4); Finkelstein, [0014] (“[T]he voltage regulator(s) 130 may be coupled to a single power plane 135 (e.g., supplying power to all the cores 106) or to multiple power planes 135 (e.g., where each power plane may supply power to a different core or group of cores).”).

214. Although Therien’s Figure 4 describes a particular example in which there is a processor power domain and a graphics power domain, a POSITA would have understood that Therien’s disclosures are not so limited. A POSITA would

have understood that Therien's disclosures are much broader. *See, e.g.,* Therien, FIG. 1 (describing a more general two-domain system). For example, a POSITA would have understood that Therien's disclosures apply equally to the case of two processor power domains. As Therien explains, "although some embodiments have been described with respect to particular components, *any other types of components* may be evaluated and/or allocated power." Therien, [0037] (emphasis added). Therien therefore discloses, or at least suggests to one ordinarily skilled in the art, other arrangements of components, including two processor domains as I discuss in the context of claim 8 for the Finkelstein-Felter combination of Ground 3. *See* Section IX.C.3.b (explaining how the combination of Finkelstein and Felter satisfy feature [8a] for Ground 3). It is therefore my opinion that Therien's teachings apply equally to the case of two processor domains and that the Finkelstein-Felter-Therien combination renders claim 12 obvious for essentially the same reasons it renders claim 15 obvious.

215. Additionally, a POSITA would have understood that Finkelstein's graphics controller could be implemented as a designated processor core (or group of processor cores) in Finkelstein's processor 102-1. In that case, the distinction I note in the preceding paragraph above would not even be a distinction at all. *See, e.g.,* Lalonde, [0023] ("[T]he use of GPU may be interchangeable with a CPU in some embodiments. For example, the processors 502 may include logic to support

graphics functions.”), FIG. 5 (disclosing the same architecture as Finkelstein’s FIG. 6). This further supports my opinion that Therien’s teachings apply equally to the case of two processor domains and that the Finkelstein-Felter-Therien combination renders claim 12 obvious for essentially the same reasons it renders claim 15 obvious.

3. Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.”

216. It is my opinion that the combined teachings of Finkelstein and Felter, further combined with Therien’s teachings, further renders claim 17 obvious. More specifically, it is my opinion that a POSITA would have found it obvious based on Therien’s disclosures to program Finkelstein’s power management logic, as adapted to perform Felter’s modified pro rata power allocation technique, to favor the graphics controller domain over the processor core domain via a preference value.

217. As I explain in the context of claim 17 for Ground 2, Finkelstein discloses that the graphics controller (second domain), which performs “Graphics Effect(s) (GFX),” may demand more power relative to the processor cores (first domain) during a “GFX intensive workload.” Finkelstein, [0026]. Accordingly, a POSITA would have understood Finkelstein to disclose distributing more power to the graphics controller during graphics-heavy workloads. *See* Finkelstein, [0026]

(teaching to “allow for use of . . . unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload”), [0050] (describing GFX functions as “high-performance”). During time intervals where such loads are present, a POSITA would have understood to prioritize these “high-performance” graphics workloads over workloads performed by the processor core (first domain) to achieve efficient graphics performance. Thus, when Finkelstein’s “user (or application defined) preferences” indicate a preference for the graphics controller power plane over the processor core power plane (such that $W_{k,2} > W_{k,1}$), Finkelstein’s power management logic would favor power allocation to the second domain over the first domain. Finkelstein, [0034] (“[A] single power plane may obtain the entire budget, so the corresponding weight vector is a unit vector. In the general case, power plane i may obtain the portion of the budget: $E^{k,i} = W_{ki} E^k$ (2) . . .”).

218. Therien discloses expressly that its power budget allocation engine (analogous to Finkelstein’s power management logic) may “determine whether to adjust a power budget allocation . . . based [on] a *pre-determined preference* (e.g., *indicating that graphics or computational performance should be given higher priority in the system* 100).” Therien, [0012] (emphasis added). Therien further explains that such determinations may be made “on a periodic basis . . . and/or upon a change in a power budget allocation adjustment request signal.” Therien, [0013]. When the system is “configured such that a graphics subsystem will typically receive

preferential power allocations as compared to a CPU,” Therien’s power budget allocation engine (analogous to Finkelstein’s power management logic) “may work together with various system components to facilitate an efficient allocation of power” based on these preferential power allocations. Therien, [0017]-[0018]. Put another way, Therien discloses that the system may be programmed to prefer the graphics domain over the CPU (processor) domain, as in Therien’s Figure 5, which I have reproduced below. *See* Therien, [0032]-[0034] (describing an example in which “it is assumed that a system includes a graphics subsystem and a CPU and that a preference for graphics performance has been established (over CPU processing performance)”), FIG. 5.

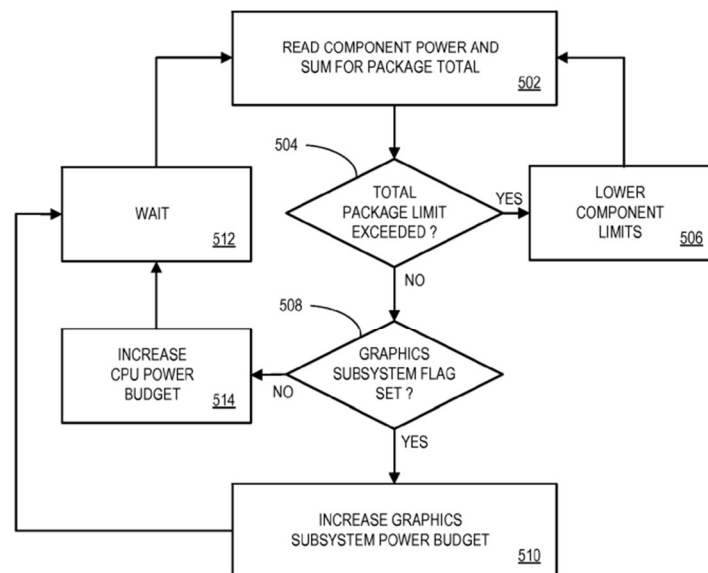


FIG. 5

Therien, FIG. 5

219. Therien's express disclosures further support my opinion that a POSITA would have been motivated to implement a preference value that favors allocating the available power/energy budget to the graphics controller domain over the processor core domain "to facilitate an efficient allocation of power" and ensure that the graphics controller has sufficient power to perform the various graphics-intensive functions demanded of it. *See* Therien, [0017]-[0018] ("[A] computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU. . . . The preference might be determined, for example, when a system is being designed or based on a selection by a user of the system (e.g., via a graphical interface). Thus, the power budget allocation engine may work together with various system components to facilitate an efficient allocation of power."). A POSITA would have understood that implementing such a preference value would have been useful when the system is tasked with performing graphics-heavy workloads associated with graphics-intensive applications. *See* Finkelstein, [0026] ("[I]t is possible to share a common package power/energy budget and allow for use of, for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.").

220. As I explain in the context of claim 17 for Ground 3, a POSITA would have reasonably expected to succeed in configuring Finkelstein's power

management logic, as modified according to Felter, to prioritize the graphics controller over the processor cores because it would not have required anything beyond ordinary and routine skill (e.g., by setting $W_{k,2} > W_{k,1}$, such that $W_{k,2}$ would represent a preference value). In fact, Finkelstein expressly contemplates that such “preferences” can be “user (or application defined),” and Therien provides additional descriptions in furtherance of that goal. Finkelstein, [0034]; *see also* Therien, [0017] (“[A] computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU.”).

E. Ground 5: The Felter-Therien Combination Renders Obvious Claim 12

- 1. Claim 12: “The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.”**

221. It is my opinion that the teachings of Felter, further combined with the teachings of Therien, renders claim 12 obvious. For essentially the same reasons I explain in the context of claim 12 for Grounds 1 and 4, the combination of Felter and Therien satisfies claim 12 because Felter’s sharing policy value ($PACT_N/PACT_{TOT}$) of a component (domain) would be incremented if its request for a higher frequency was not granted. *See* Section IX.A.2 above (describing Felter’s

disclosures in the context of claim 12), Section IX.D.2 above (explaining how the Finkelstein-Felter-Therien combination renders claim 12 obvious); Therien, [0019]-[0022] (describing how and why “a computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU”). It is my opinion that a POSITA would have found it obvious to configure Felter’s power administrator (analogous to Finkelstein’s power management logic and Therien’s power budget allocation engine) to implement Therien’s graphics preference value for the same reasons it would have been obvious to incorporate it into Finkelstein’s power management logic, as adapted to perform Felter’s modified pro rata power allocation technique. *See* Section IX.D.2 above (explaining how the Finkelstein-Felter-Therien combination renders claim 12 obvious). The only difference is that, instead of setting $W_{k,2} > W_{k,1}$ as in the Finkelstein-Felter-Therien combination, a POSITA would have found it obvious in the Felter-Therien combination to set $(PACT_1/PACT_{TOT}) > (PACT_2/PACT_{TOT})$. This difference, however, is immaterial since these $W_{k,i}$ values and $(PACT_N/PACT_{TOT})$ values are mathematically equivalent. *See* Section IX.C.2 above (explaining this equivalence).

F. Ground 6: The Finkelstein-Therien Combination Renders Obvious Claims 15 and 17

- 1. Claim 15: “The system of claim 13, wherein the power sharing logic is to increment the first power sharing value when a request for a higher frequency for the first domain is not granted, and to increment the second power sharing value when a request for a higher frequency for the second domain is not granted.”**

222. It is my opinion that the teachings of Finkelstein, further combined with the teachings of Therien, renders claim 15 obvious. For the same reasons I provide regarding claim 15 in the context of Ground 4 (the Finkelstein-Felter-Therien combination), the combination of Finkelstein and Therien satisfies claim 15 because the sharing policy value (W_{ki}) of a power plane (domain) would be incremented if its request for a higher frequency was not granted. *See* Finkelstein, [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of, for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”), [0034] (“[A] single power plane may obtain the entire budget, so the corresponding weight vector is a unit vector. In the general case, power plane i may obtain the portion of the budget: $E^{k,i}=W_{ki}E^k(2) \dots$ ”).

223. Therien explains that adjusting the power budget allocation for a component may be appropriate if “an additional allocation of power could be usefully consumed.” Therien, [0019]. For example, “the graphics subsystem might

evaluate a current frame rate, render p-state, render frequency, and/or application being executed by the graphics subsystem (e.g., a graphics-intensive 3D game) before requesting an increase in its power allocation. Thereien, [0020]. Or a CPU “might evaluate an instruction execution rate and/or an application being executed by the CPU before requesting an increase in its power allocation.” Therien, [0020]. If additional power allocation would be useful, Therien explains that “an increase request flag can be raised (or lowered),” and the system may experience an “increase[d] . . . frame rate” or an “increase[e] [in] the number of instructions retired per second.” Therien, [0021]-[0022]. A POSITA would have understood that such results are achieved by increasing the component’s operating frequency. For example, Therien’s power budget allocation engine may “adjust a power budget allocation signal provided to the . . . component” so that “an individual *power limit imposed on a component* . . . may be relaxed or *raised* as appropriate.” Therien, [0013]-[0016] (emphasis added).

224. The only difference regarding how the Finkelstein-Therien combination applies to claim 15 (as compared to how the Finkelstein-Felter-Therien combination applies to claim 15) is that a POSITA would have understood to increment Finkelstein’s power sharing values (W_{ki}) according to Finkelstein’s pro rata allocation technique rather than Felter’s modified pro rata technique. *Compare* Finkelstein, [0034], *with* Felter, [0061], FIG. 5; *see also* Section IX.C.2 above

(describing the combination of Finkelstein and Felter). That difference, however, is immaterial here. Incrementing W_{ki} in the Finkelstein-Therien combination has the same effect as incrementing $W_{k,i}$ in the Finkelstein-Felter-Therien combination since both quantities represent the power sharing values in their respective combinations.

2. Claim 17: “The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.”

225. It is my opinion that the teachings of Finkelstein, further combined with the teachings of Therien, also renders claim 17 obvious. For the same reasons I provide regarding claim 17 in the context of Ground 4 (the Finkelstein-Felter-Therien combination), the combination of Finkelstein and Therien satisfies claim 17 because a POSITA would have found it obvious to program Finkelstein’s power management logic to employ a preference value that favors the second domain (graphics controller power plane) over the first domain (processor core power plane). *See* Finkelstein, [0026] (“[I]t is possible to share a common package power/energy budget and allow for use of, for example, unused processor core power for more Graphics Effect(s) (GFX) performance in a GFX intensive workload, when processor resources are not fully utilized.”), [0034] (“[A] single power plane may obtain the entire budget, so the corresponding weight vector is a unit vector. In the general case, power plane i may obtain the portion of the budget: $E^{k,i}=W_{ki}E^k$ (2)

. . . .”); Therien, [0019]-[0022] (describing how and why “a computing system might be configured such that a graphics subsystem will typically receive preferential power allocations as compared to a CPU”), FIG. 3.

226. The only difference regarding how the Finkelstein-Therien combination applies to claim 17 (as compared to how the Finkelstein-Felter-Therien combination applies to claim 17) is that a POSITA would have understood to set Finkelstein’s power sharing values (W_{ki}) according to Finkelstein’s pro rata allocation technique rather than Felter’s modified pro rata technique. *Compare* Finkelstein, [0034], *with* Felter, [0061], FIG. 5; *see also* Section IX.C.2 above (describing the combination of Finkelstein and Felter). That difference, however, is immaterial here. Setting $W_{k2} > W_{k1}$ in the Finkelstein-Therien combination has the same effect as setting $W_{k,2} > W_{k,1}$ in the Finkelstein-Felter-Therien combination in the sense that W_{k2} still favors power allocation to the graphics controller over the processor cores and thus still acts as a preference value as claim 17 recites.

X. CONCLUSION

227. For the reasons set forth above, it is my opinion that a person ordinarily skilled in the art would have found claims 8 and 12-17 of the ’316 patent obvious.

228. In signing this declaration, I understand that it will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I acknowledge that I may be subject to cross-

examination in this case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for cross-examination within the United States during the time allotted for cross-examination.

229. I declare that all statements made herein of my knowledge are true, and that all statements made on information and belief are believed to be true, and that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Date: January 6, 2025

Respectfully submitted,



Murali Annavaram, Ph.D.

LISTING OF CHALLENGED CLAIMS

Claim 8	
[8Pre]	A method comprising:
[8a]	determining, in a power controller of a multi-domain processor, a power budget for the multi-domain processor for a current time interval, the multi-domain processor including at least a first domain and a second domain;
[8b]	determining, in the power controller, a portion of the power budget to be allocated to the first and second domains, including
[8c]	allocating a minimum reservation value to the first domain and a minimum reservation value to the second domain, and
[8d]	sharing a remaining portion of the power budget according to a first sharing policy value for the first domain and a second sharing policy value for the second domain; and
[8e]	controlling a frequency of the first domain and a frequency of the second domain based on the allocated portions.
Claim 12	
The method claim 8, wherein the first sharing policy value is to be incremented when a request for a higher frequency for the first domain is not granted, and the second sharing policy value is to be incremented when a request for a higher frequency for the second domain is not granted.	

Claim 13	
[13Pre]	A system comprising
[13a]	a multicore processor having a first domain including a plurality of cores, a second domain including a graphics engine, and a third domain including system agent circuitry,
[13b]	the third domain to operate at a fixed power budget and to dynamically allocate a variable power budget between the first and second domains, wherein
[13c]	the system agent circuitry includes a power sharing logic to determine the variable power budget for a current time interval and
[13d]	to allocate a first portion of the variable power budget to the first domain according to a first power sharing value for the first domain, and to allocate a second portion of the variable power budget to the second domain according to a second power sharing value for the second domain; and
[13e]	a dynamic random access memory (DRAM) coupled to the multicore processor.
Claim 14	
The system of claim 13, wherein the power sharing logic is to dynamically allocate substantially all of the variable power budget to the first domain for a first workload, and to dynamically allocate substantially all of the variable power budget to the second domain for a second workload executed after the first workload.	
Claim 15	
The system of claim 13, wherein the power sharing logic is to increment the first power sharing value when a request for a higher frequency for the first domain is not granted, and to increment the second power sharing value when a request for a higher frequency for the second domain is not granted.	

Claim 16
The system of claim 13, wherein the power sharing logic is to further allocate the first portion of the variable power budget according to a first minimum reservation value for the first domain and allocate the second portion of the variable power budget according to a second minimum reservation value for the second domain.
Claim 17
17. The system of claim 13, wherein the power sharing logic is to further allocate the variable power budget according to a preference value, the preference value to favor the second domain over the first domain.