

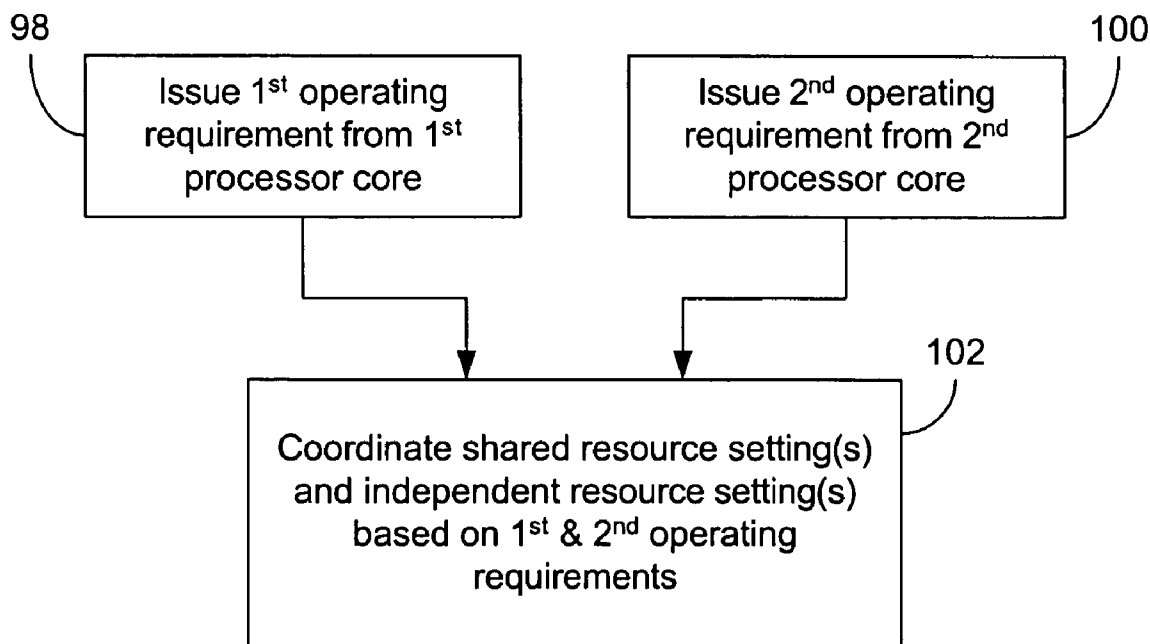


US 20060026447A1

(19) **United States**(12) **Patent Application Publication****Naveh et al.**(10) **Pub. No.: US 2006/0026447 A1**(43) **Pub. Date: Feb. 2, 2006**(54) **POWER MANAGEMENT COORDINATION
IN MULTI-CORE PROCESSORS**(52) **U.S. Cl. 713/322**(75) **Inventors: Alon Naveh, Ramat Hasharon (IL);
Efraim Rotem, Haifa (IL); Eliezer
Weissmann, Haifa (IL)**(57) **ABSTRACT**

Correspondence Address:
INTEL CORPORATION
P.O. BOX 5326
SANTA CLARA, CA 95056-5326 (US)

Systems and methods of managing power provide for issuing a first operating requirement from a first processor core and issuing a second operating requirement from a second processor core. In one embodiment, the operating requirements can reflect either a power policy or a performance policy, depending upon the factor that is currently most important to software. Hardware coordination logic is used to coordinate a shared resource setting with the operating requirements. The hardware coordination logic is also able to coordinate the shared resource setting with independent resource settings of the first and second processor cores based on the operating requirements.

(73) **Assignee: INTEL CORPORATION**(21) **Appl. No.: 10/899,674**(22) **Filed: Jul. 27, 2004****Publication Classification**(51) **Int. Cl.**
G06F 1/26 (2006.01)**96**

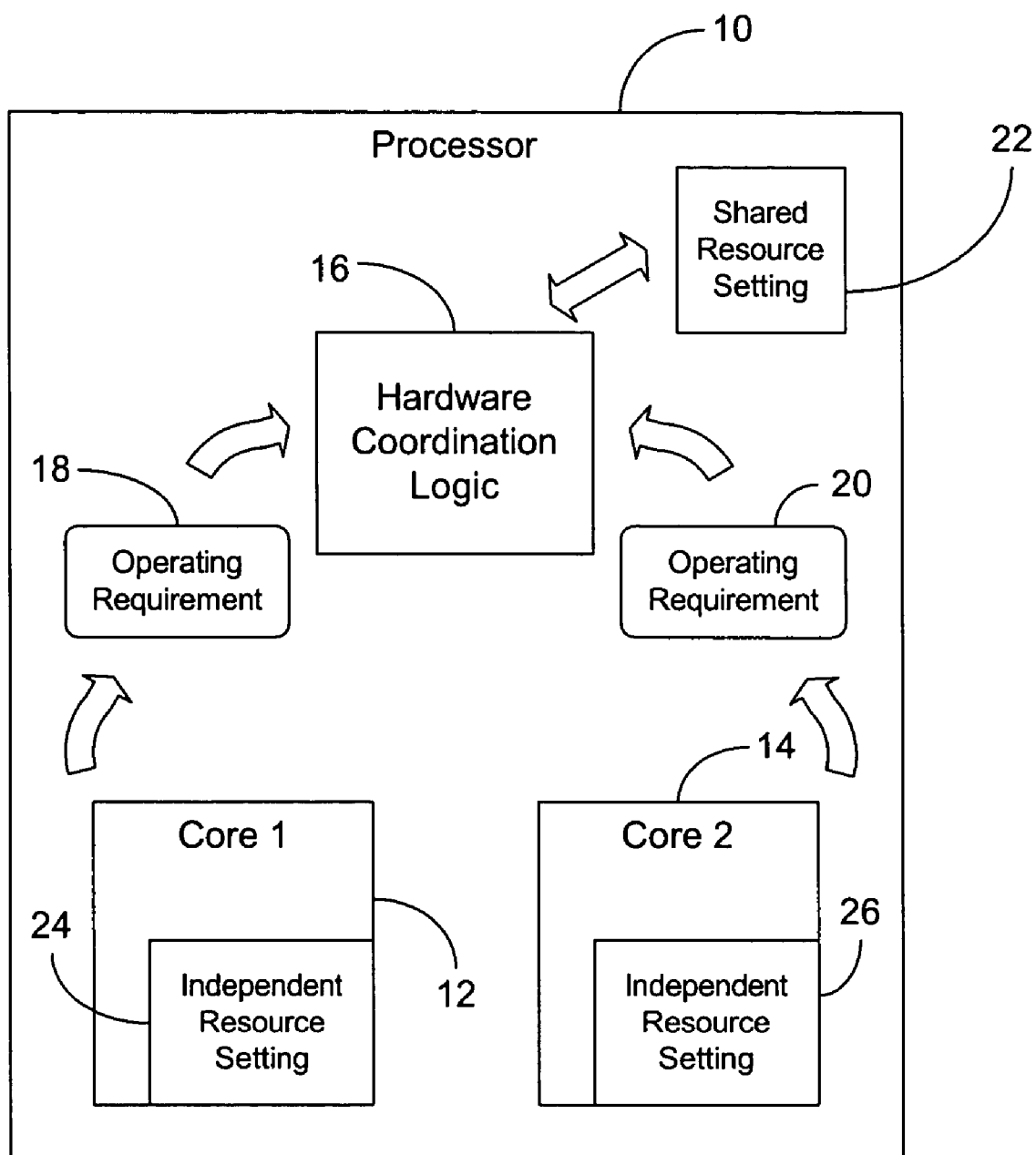
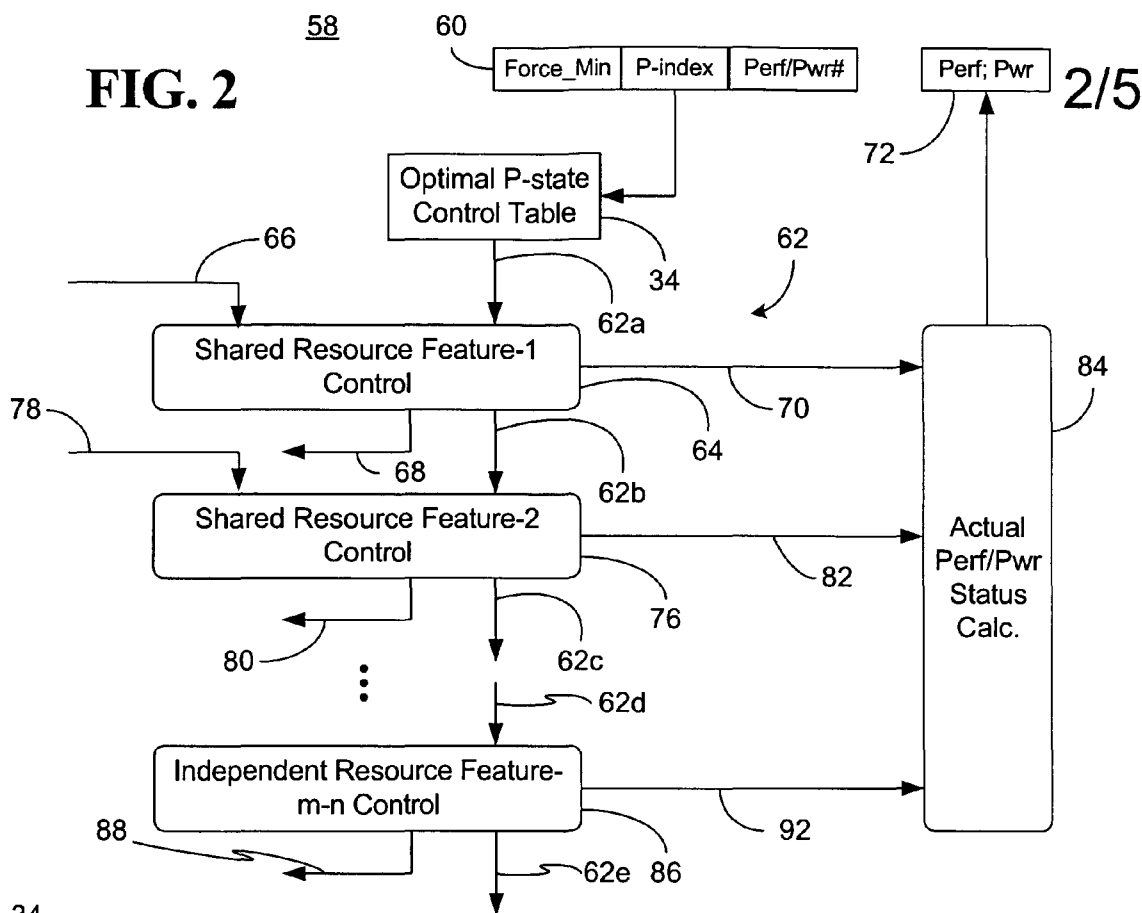


FIG. 1

FIG. 2

34

Combined Optimal State				Shared Feature SpeedStep® Control				Independent Feature TM1 Control		
Pindex	Perf	Abs-Pwr	Pwr%	Frequency	Vcc	Pwr%	Perf	Throttle%	Pwr%	Perf
0	1	100	100%	1700	1.233	100%	100%	0	1	100%
1	0.97	84	84%	1600	1.183	84%	97%	0	1	100%
2	0.94	68	68%	1500	1.133	68%	94%	0	1	100%
3	0.91	54	54%	1400	1.083	54%	91%	0	1	100%
4	0.88	41	41%	1300	1.008	41%	88%	0	1	100%
5	0.85	28	28%	1200	0.933	28%	85%	0	1	100%
6	0.82	16	16%	1100	0.85	16%	82%	0	1	100%
7	0.79	15	15%	1000	0.85	15%	79%	0	1	100%
8	0.76	13	13%	900	0.85	13%	76%	0	1	100%
9	0.73	12	12%	800	0.85	12%	73%	0	1	100%
10	0.7	10	10%	700	0.85	10%	70%	0	1	100%
11	0.67	9	9%	600	0.85	9%	67%	0	1	100%
12	0.58625	8	8%	600	0.85	9%	67%	0.125	0.9063	88%
13	0.5025	7	7%	600	0.85	9%	67%	0.25	0.8125	75%
14	0.41875	6	6%	600	0.85	9%	67%	0.375	0.7188	63%
15	0.335	6	6%	600	0.85	9%	67%	0.5	0.625	50%
16	0.1675	4	4%	600	0.85	9%	67%	0.75	0.4375	25%

50 52 42 44 36 38 40

FIG. 3

FIG. 4

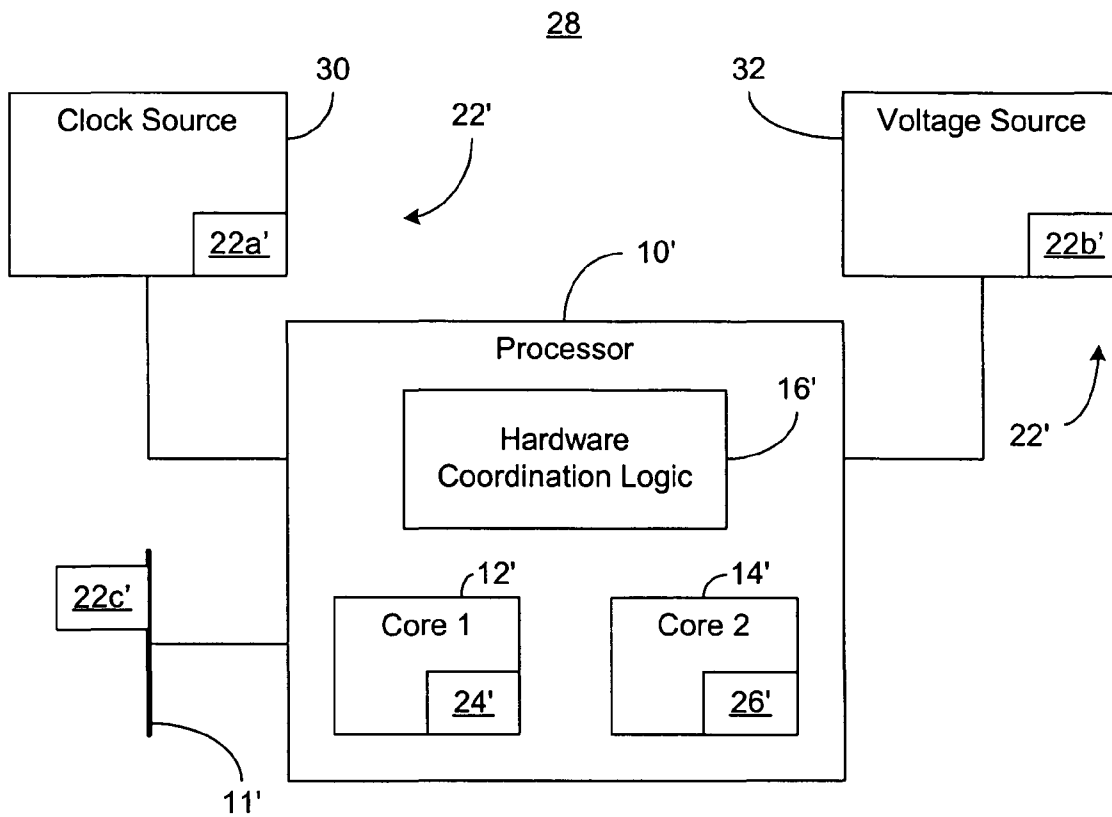
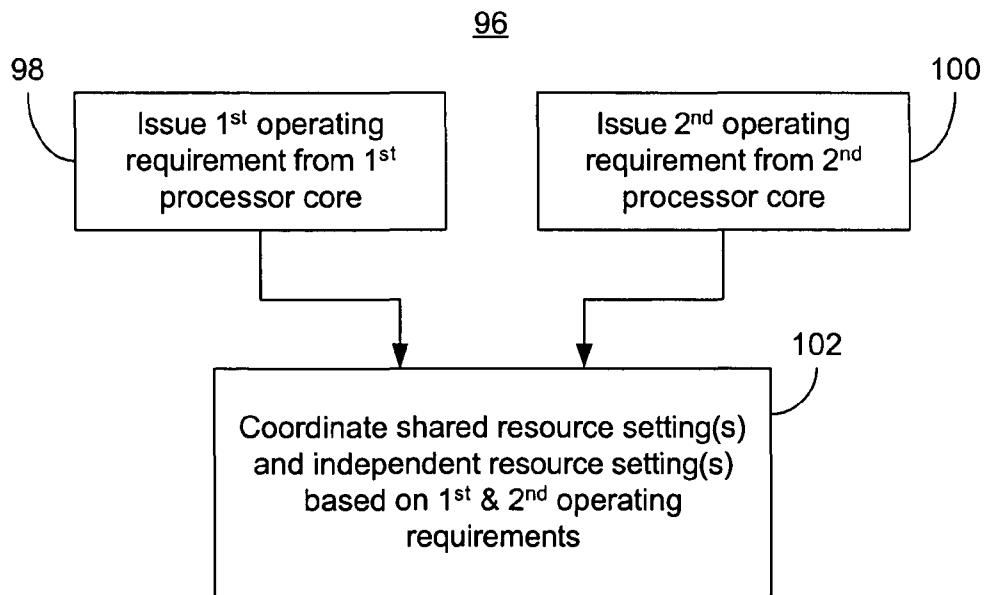


FIG. 5

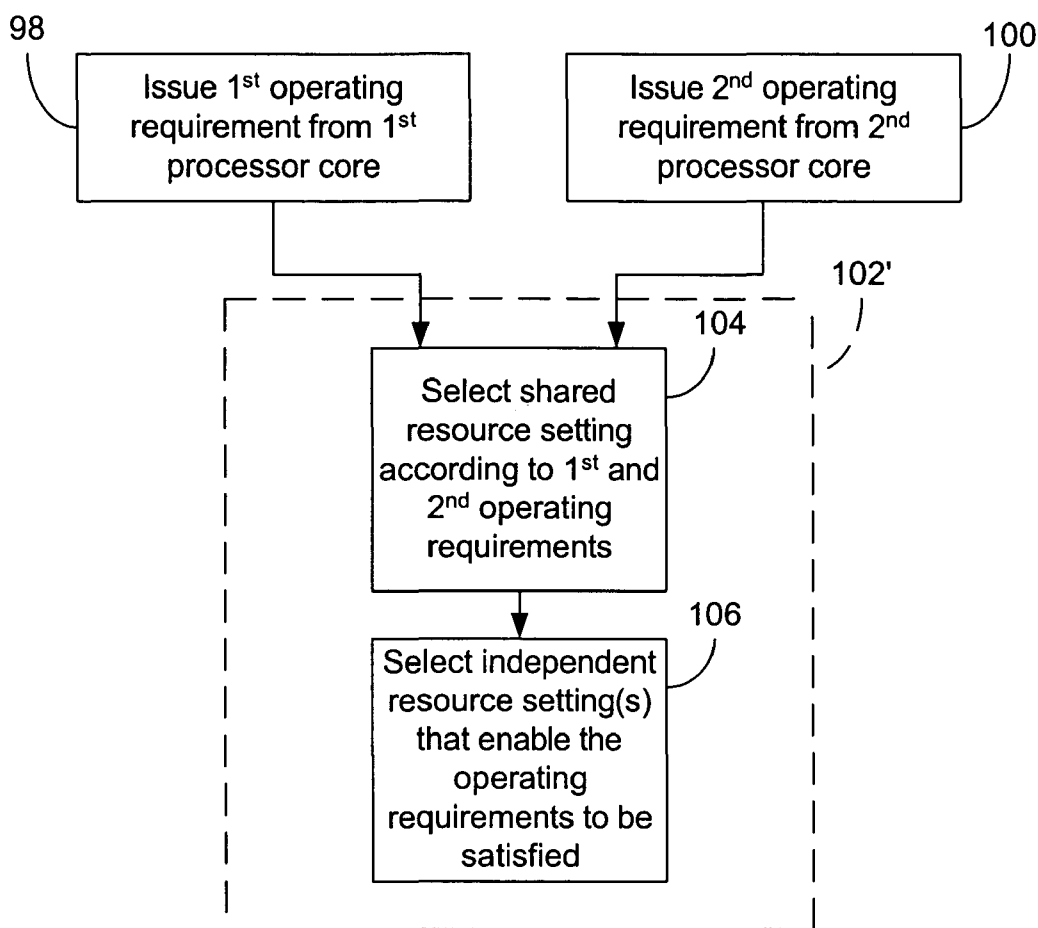
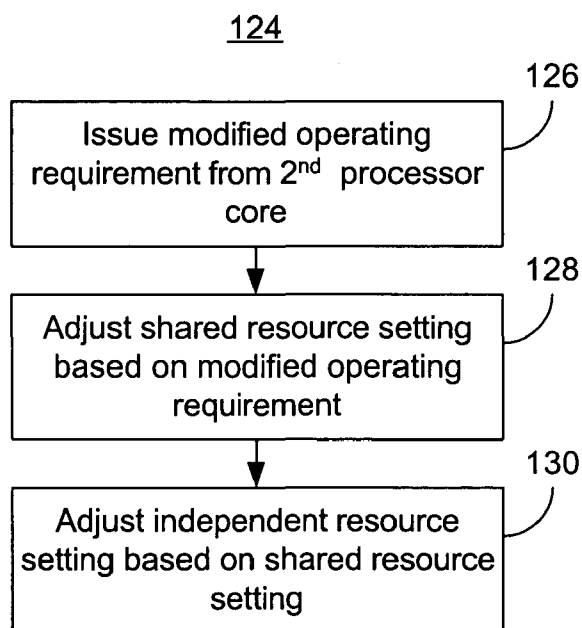


FIG. 6

FIG. 8



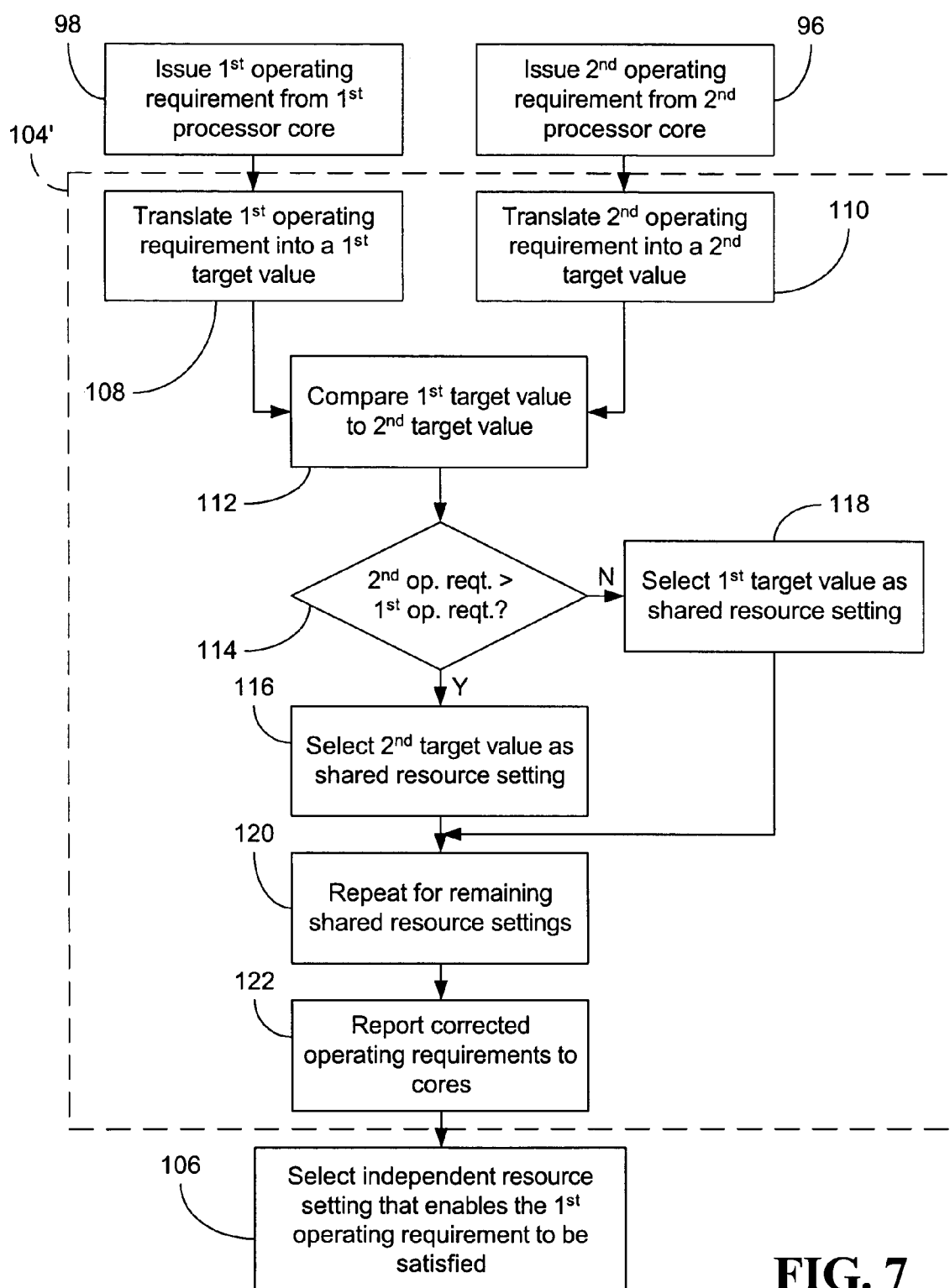


FIG. 7

POWER MANAGEMENT COORDINATION IN MULTI-CORE PROCESSORS

BACKGROUND

[0001] 1. Technical Field

[0002] One or more embodiments of the present invention generally relate to power management. In particular, certain embodiments relate to coordinating power management activities in multi-core processors.

[0003] 2. Discussion

[0004] As the trend toward advanced processors with more transistors and higher frequencies continues to grow, computer designers and manufacturers are often faced with corresponding increases in power consumption. Furthermore, manufacturing technologies that provide faster and smaller components can at the same time result in increased leakage power. Particularly in mobile computing environments, increased power consumption can lead to overheating, which may negatively affect performance, and can significantly reduce battery life.

[0005] Some modern mobile computing systems address the concern over reduced battery life by implementing processor-based power management schemes. For example, one of the more popular approaches is to dynamically lower processor power consumption by scaling down the core voltage and clock frequency of the processor when high performance is not needed or desired. Power management can also be enhanced by scaling the frequency of the bus that the processor uses to communicate with other components such as chipsets, memory subsystems, input/output (I/O) devices, etc. Another approach is to switch, or “gate”, the clock to the processor on and off (i.e., “throttle” the clock) to achieve power savings. Yet another example of processor-based power management is to throttle architectural components of the processor such as internal arrays, execution units, and so on. Some techniques make use of various combinations of these approaches to further reduce power consumption. While these power management schemes have been acceptable under certain circumstances, a number of concerns remain.

[0006] One concern relates to recent trends toward more complex processor architectures, which can include multiple processors in a system, with multiple cores on each processor. In such a case, some of the power-related resources and associated controls may be shared among processor cores, where other resources and controls may be dedicated to a given core. The dedicated resources and controls are independent from the resources and controls of other cores, where the shared resources and controls depend upon the state of each of the cores. Conventional power management schemes, however, may not work well in these cases because they are structured for single-processor environments.

[0007] For example, one such scheme controls transitions of the processor between various power states at the operating system (OS) level. Although future OS implementations may be able to manage power on a per core basis, it is not at all clear whether the OS will be able to coordinate between the cores. Furthermore, even if OS implementations were to achieve per core power management, efficiency may still be an issue of concern. In particular, the OS may not be aware of all of the available performance/power control

mechanisms and may not be able to balance the trade-offs between them. In addition, relying on software to coordinate such complicated architectures could increase OS overhead and contribute to software calculation complexity. Software-based power management of multi-core processors may also be difficult to implement from a timing standpoint because of the rapid state changes that can occur in inter-dependent cores.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] The various advantages of the embodiments of the present invention will become apparent to one skilled in the art by reading the following specification and appended claims, and by referencing the following drawings, in which:

[0009] FIG. 1 is a block diagram of an example of a processor according to one embodiment of the invention;

[0010] FIG. 2 is a diagram of an example of a power management architecture according to one embodiment of the invention;

[0011] FIG. 3 is a diagram of an example of a performance state transition table according to one embodiment of the invention;

[0012] FIG. 4 is a block diagram of an example of a system according to one embodiment of the invention;

[0013] FIG. 5 is a flowchart of an example of a method of managing power according to one embodiment of the invention;

[0014] FIG. 6 is a flowchart of an example of a process of coordinating a shared resource setting with an independent resource setting according to one embodiment of the invention;

[0015] FIG. 7 is a flowchart of an example of a process of selecting a shared resource setting according to one embodiment of the invention; and

[0016] FIG. 8 is a flowchart of an example of a process of adjusting resource settings according to one embodiment of the invention.

DETAILED DESCRIPTION

[0017] FIG. 1 shows a processor 10 having a first core 12, a second core 14 and hardware coordination logic 16 operatively coupled to the first core 12 and the second core 14. The terms “first” and “second” are used herein for ease of discussion only. Furthermore, while a dual-core configuration is shown here, the number of cores in the processor 10 can readily be increased without parting from the spirit and scope of the embodiments of the invention. Each of the illustrated cores 12, 14 are fully functional as logical processors and include traditional execution units, level one (L1) caches, etc. The illustrated dual-core configuration can therefore provide substantial performance benefits over conventional single-core processors.

[0018] The processor 10 has a number of features/resources that enable a significant reduction in power. For example, the processor 10 may have a feature that is able to scale down the processor clock frequency and power supply voltage (e.g., core voltage) when high performance is either not needed or not desired. Another power saving feature might provide for scaling the clock frequency of a bus (not

shown). Yet another feature could provide for throttling (or “gating”) the clocks off and on, which reduces power consumption. Clock throttling can be conducted at the package level or to individual portions of the processor 10. Yet another power saving feature might involve throttling architectural components of the cores 12, 14 in order to reduce power and/or decrease temperature.

[0019] While each of these features can be quite beneficial from a power management standpoint, some of them make use of resources that are shared by the cores 12, 14, and others concern resources that are independently controlled by the cores 12, 14. The illustrated processor 10 generally has an architecture that is able to dynamically support both scenarios, and provides a number of advantages over conventional power management schemes.

[0020] For example, the illustrated processor 10 may have a processor clock frequency setting and a power supply voltage setting that are shared by both cores 12, 14, whereas an independent clock throttle setting may control how often the clock is applied to the first core 12 only. Thus, the clock frequency and power supply voltage settings could be viewed as a shared resource setting 22 and the independent clock throttle setting could be viewed as an independent resource setting 24. Similarly, the second core 14 may also have an independent clock throttle setting that is represented by an independent resource setting 26.

[0021] In addition to independent clock throttling, the cores 12, 14 may be able to independently throttle various functional blocks of their internal architectures. For example, the independent resource settings 24, 26 could provide for the throttling of architectural components such as internal arrays, reorder buffers (ROBs), reservation station (RS) tables, parallel units, execution units, and so on.

[0022] In the illustrated embodiment, the first core 12 issues a first operating requirement 18 and the second core 14 issues a second operating requirement 20. The coordination logic 16 can coordinate the shared resource setting 22 with the operating requirements 18, 20. As will be discussed in greater detail below, the operating requirements 18, 20 could be generated according to a policy such as a power policy or a performance policy, and may be a product of the operating system (OS) running on each of the cores 12, 14. For example, the first core 12 might have user/software selectable power goals that allow it to reduce power consumption by twenty percent. In this case, the first operating requirement 18 could be a power requirement that indicates an eighty percent power level. Alternatively, the second core 12 might be in a mode of operation that requires full power so that the second operation requirement 20 indicates a one hundred percent power level. Specific values are used by way of example, only.

[0023] The operating requirements 18, 20 could also identify a performance requirement such as a performance level or an index into a performance state transition table, as discussed in greater detail below. By enabling the hardware coordination logic 16 to function based on either a performance or a power policy, the processor 10 is able to focus on what is currently the most important factor for software to control, while perhaps compromising the other. Although using performance/power policies can be helpful when coordinating across multiple features and a common policy (performance or power) is needed, other approaches may be

used. For example, incorporating the actual resource settings into the operating requirements may also be an acceptable approach.

[0024] As already noted, the coordination logic 16 can select the shared resource setting 22 according to the first operating requirement 18 and the second operating requirement 20. The result is an operating state that may or may not be meet the operating requirements 18, 20. If, for example, the first operating requirement 18 is not satisfied by the shared resource setting 22, the first core 12 selects a modified/adjusted value for the first independent resource setting 24 that enables the first operating requirement 18 to be satisfied. Alternatively, if the second operating requirement 20 is not satisfied by the shared resource setting 22, the second core 14 selects a modified/adjusted value for the second independent resource that enables the second operating requirement 20 to be satisfied. The coordination logic 16 may therefore determine the resulting independent resource settings 24, 26, based on the difference between the individual operating requirements 18, 20, and the shared resource setting 22, which reflects the actual achieved coordinated state. Simply put, the independent resource settings 24, 26 are able to compensate for any power/performance deficiencies associated with the shared resource setting 22, which may be constrained by one or more of the cores 12, 14.

[0025] Thus, in the above example of mismatched operating requirements, the coordination logic 16 would take into consideration the fact that the second core 14 requires a higher power level, and would select the coordinated shared resource setting 22 to satisfy this requirement. The shared resource setting 22 would therefore result in an operating state of one hundred percent power for both cores 12, 14, and the first core 12 would still have an operating requirement 18 of eighty percent. Since the first core 12, has not yet satisfied the first operating requirement 18, the first core 12 selects a corrected/adjusted value for the first independent resource setting 24 that enables the first operating requirement 18 to be satisfied. For example, the first core 12 might increase its independent clock throttle setting to achieve the desired twenty percent reduction in power. Thus, by coordinating the shared resource setting 22 with the independent resource settings 24, 26, the coordination logic 16 is able to manage power efficiently in a multi-core environment.

[0026] The coordination logic 16 can select the shared resource setting 22 by translating the first operating requirement 18 into a first target value and translating the second operating requirement 20 into a second target value, where the target values represent potential values for the shared resource setting 22. The coordination logic 16 may then compare the first target value to the second target value and select the higher of the two as the shared resource setting 22. It should also be noted that, the coordination logic 16 could simply compare the operating requirements 18, 20 and select the higher of the two. The coordination logic 16 is therefore able to function as a “maximum detector” for operating requirements and/or target values of the shared resource setting 22.

[0027] Alternatively, the coordination logic 16 may select the lesser value if a “minimum-of-both-requests” coordination policy is appropriate. Such a situation might occur in the case of shared clock throttling or if the OS decides to set a

“Force_Min” flag in the controls, where the Force_Min flag could notify the coordination logic 16 that it needs to enforce a MINIMUM operating point policy, ensuring that the minimum of both cores’ individual throttling request is the one selected. For simplicity, some of the examples described herein will refer to the coordination policy as a maximum. It should be noted, however, that the advantages associated with the embodiments of the invention can also apply to minimum policies.

[0028] If the second core 14 subsequently issues a modified operating requirement that corresponds to a reduced target value, the coordination logic 16 is able to adjust the shared resource setting 22 based on the modified operating requirement. In such a case, the first core 12 could adjust the independent resource setting 24 based on the adjusted shared resource setting. In the above example of mismatched operating requirements, the first core 12 might reduce its independent clock throttle setting (because it is no longer needed). It should be noted that by implementing the coordination logic in hardware, fast OS response times in order to accommodate rapid state changes are not imposed on the system. It should also be noted that if the independent resource settings 24, 26 are known to be more efficient than the shared resource setting 22, the above-described process may be reversed so that the independent resource settings 24, 26 are selected first, with the shared resource setting 22 compensating for the independent resource settings 24, 26.

[0029] FIGS. 2 and 3 show a power management architecture 58 that can be used to implement hardware coordination logic and a performance state transition table 34 that can be used to facilitate the above-described translation, respectively. While the architecture 58 and the table 34 can be very helpful in managing power, other approaches may be used. In the illustrated example, a performance state (“P-state”) activation command 60, which defines operating requirement characteristics such as a force minimum characteristic, performance index, performance level and/or power level for each processor core is used to access the performance state transition table 34. Although only a single activation command 60 is shown, multiple activation commands can be applied to the table 34 simultaneously.

[0030] In the illustrated example, the operating requirements can be defined as power requirements such as a combined optimal state absolute power (“Abs-Pwr”) 42 or a power percentage (“Pwr %”) 44. The operating requirements could also be defined as performance requirements such as a performance index (“Pindex”) 50 or a combined optimal state absolute performance (“Perf”) 52. To support relatively complex hardware coordination, it can be seen that shared resource settings may be integrated across various features and the actual operating requirements may be specified at a “global” level. Thus, in the illustrated example, the activation commands 60 include operating requirements that refer to the combined optimal state.

[0031] As already noted, the target values for the resource settings can be identified by feature, where the Speedstep® feature (“Shared Feature”) may provide control over shared resource settings, which are illustrated as clock frequency settings (“Frequency”) 36 and core voltage settings (“V_{cc}”) 38. The TM1 feature (“Independent Feature”), on the other hand, may provide control over independent resource settings, which are illustrated as independent clock throttle settings (“Throttle %”) 40. As the operating requirements and target values are received by the coordination logic, they can be compared to one another for selection of an appropriate global setting.

[0032] In another particular example, if the first core requests an 88% performance level, the coordination logic can use the 88% performance “operating requirement” to index into the combined optimal state portion of the table. A target frequency/voltage value of 1300 MHz/1.008V can be extracted from the table and placed on a control bus 62 (62a-62e) for delivery to a first shared resource control block 64. The illustrated first shared resource control block 64 therefore corresponds to the shared feature shown in the table 34. Simultaneously with the first core request, the second core may issue an operating requirement such as a request to operate at the 100% performance level. The coordination logic can therefore use the 100% performance to index into the combined optimal state portion of the table, extract the target frequency/voltage value of 1700 MHz/1.233V from the table, and place it on the control bus 66.

[0033] If the illustrated system is in the “max detect” mode (i.e., Force_Min is not set), the first shared resource control block 64 compares the two operating requirements and determines that the appropriate global setting is 100% performance. Accordingly, the target frequency/voltage value of 1700 MHz/1.233V may be selected as the shared resource setting 68. In the illustrated example, the shared resource setting 68 is applied to the clock source and/or voltage source, instructing the shared resources to operate at the appropriate frequency and/or core voltage defined by the shared resource setting 68. The first shared resource control block 64 also sends an operating state signal 70 to status calculation logic 84, which generates an actual operating state calculation 72. The operating state signal 70 identifies the performance and/or power state resulting from the shared resource setting 68. Thus, in this example the operating state signal 70 would identify the 100% performance level.

[0034] In addition, the first shared resource control block 64 can generate corrected operating requirements and/or target values based on the shared resource setting 68 and place the corrected target values on the control bus 62b, where the corrected target values enable a second shared resource control block 76 to determine a shared resource setting 80 for the next shared resource. The corrected values correspond to the selected operating level and can be extracted from a table such as the table 34 (although only one shared feature is illustrated in the table). In the illustrated example, the shared resource setting 80 could be a bus frequency setting. The second shared resource control block 76 also collects operating requirements and/or target values of one or more other processor cores from a control bus 78 and determines the maximum in the set. This maximum represents the shared resource setting 80, which instructs the clock to operate at the appropriate shared resource setting. An operating state signal 82 is also sent to the status calculation logic 84. If additional resource control blocks are needed, corrected operating requirement and/or target values are generated and placed on the control bus 62c.

[0035] When the shared resource settings have all been completed, the power management architecture 58 provides for the use of an independent resource control block 86 to generate an independent resource setting 88, where the independent resource setting 88 enables the processor core in question to achieve any additional power savings reflected in a corrected operating requirement on control bus 62d. The corrected operating requirement can be obtained from the last of the shared resource control blocks. In the above example of a first core in need of an 88% performance level, the independent resource control block 86 might select an

independent resource setting of 0.125 independent clock throttle, which provides 88% performance. The illustrated global setting of P12 also includes a 67% performance reduction through the shared voltage/freq setting, which if combined with the 0.125 independent clock throttle gives you 57% performance overall. However, the approach described herein compensates for the performance number by using the independent resource setting when the shared resource has been already selected and fixed. The independent resource control block **86** is also able to generate an operating state signal **92** for use in generating the actual operating state calculation **72**. If the independent resource setting **88** is not able to fully satisfy the corrected operating requirement, another corrected operating requirement can be sent to the next independent resource control block (not shown) on control bus **62e**.

[0036] Turning now to **FIG. 4**, one example of a system **28** having a clock source **30** with a shared frequency setting **22a'**, a bus **11** with a shared frequency setting **22c'**, a voltage source **32** with a shared voltage setting **22b'** and a multi-core processor **10'**. The system **28** could be part of a mobile computing system such as a notebook personal computer (PC), a personal digital assistant (PDA), a wireless "smart" phone, etc., in which power consumption and overheating are issues of particular concern. The illustrated processor **10'** has a first core **12'**, a second core **14'** and hardware coordination logic **16'**. The first core **12'** issues a first operating requirement (not shown) and the second core **14'** issues a second operating requirement (not shown). The coordination logic **16'** coordinates the shared frequency settings **22'** (**22a'-22c'**) with the operating requirements and with the independent resource settings **24'**, **26'** of the first and second cores **12'**, **14'**, respectively.

[0037] **FIG. 5** shows a method **96** of managing power. The method **96** can be implemented in a multi-core processor using any available hardware and/or software programming technique. For example, one or more portions of the method **96** can be embodied in fixed functionality hardware, an application specific integrated circuit (ASIC), in a set of microcode instructions to be stored on a machine readable medium, or any combination thereof. In particular, the illustrated method **96** provides for issuing a first operating requirement from a first processor core at processing block **98**. A second operating requirement is issued from a second processor core at block **100**. Block **102** provides for coordinating a shared resource setting with the operating requirements. The shared resource setting is already coordinated with an independent resource setting based on the first operating requirement and the second operating requirement. The independent resource setting could be dedicated to the first processor core or the second processor core.

[0038] Turning now to **FIG. 6**, one approach to coordinating a shared resource setting is shown in greater detail at block **102'**. In particular, block **104** provides for selecting the shared resource setting according to the first operating requirement and the second operating requirement. Block **106** provides for selecting adjusted values for the independent resource settings that enable the operating requirements to be satisfied.

[0039] **FIG. 7** shows one approach to selecting a shared resource setting in greater detail at block **104'**. In the illustrated example, the first operating requirement is translated into a first target value at block **108** and the second operating requirement is translated into a second target value at block **110**. As already discussed, the translation at

blocks **108** and **110** can be facilitated by the use of a power state transition table. Block **112** provides for comparing the first operating requirement to the second operating requirement. Alternatively, the target values themselves can be compared. If it is determined at block **114** that the second operating requirement is greater than the first operating requirement, block **116** provides for selecting the second target value as the shared resource setting. Otherwise, the first target value is selected as the shared resource setting at block **118**. Block **120** provides for repeating the shared resource setting selection process for the remaining shared resources, if necessary. A corrected operating requirement that results from the shared resource settings is reported to the cores at block **122**.

[0040] Turning now to **FIG. 8**, a process **124** of updating shared resource settings and independent resource settings is shown. In the illustrated example, the second processor core has previously required a higher level of performance than the first processor core. Accordingly, the first processor core is using independent resource settings to compensate the shared resource setting. Block **126** provides for issuing a modified operating requirement from the second processor core. The shared resource setting is adjusted at block **128** based on the modified operating requirement. Block **130** provides for adjusting the independent resource settings of the first processor core based on the adjusted shared resource setting.

[0041] Thus, the embodiments described herein provide a number of unique advantages for power management in a multi-core processor. For example, coordinating power management in hardware coordination logic alleviates software overhead and calculation concerns. Furthermore, hardware coordination is relatively fast and obviates response time concerns associated with OS-coordinated power management. In addition, by managing power based on either a performance or a power policy, the processor is able to focus on what is currently the most important factor for software to control, while perhaps compromising the other.

[0042] Those skilled in the art can appreciate from the foregoing description that the broad techniques of the embodiments of the present invention can be implemented in a variety of forms. Therefore, while the embodiments of this invention have been described in connection with particular examples thereof, the true scope of the embodiments of the invention should not be so limited since other modifications will become apparent to the skilled practitioner upon a study of the drawings, specification, and following claims.

What is claimed is:

1. A processor comprising:

a plurality of cores to issue a corresponding plurality of operating requirements; and

coordination logic coupled to the plurality of cores, the coordination logic to coordinate a shared resource setting of the plurality of cores with the plurality of operating requirements.

2. The processor of claim 1, wherein the coordination logic is to coordinate the shared resource setting with an independent resource setting of at least one of the plurality of cores.

3. The processor of claim 2, wherein the coordination logic is to select the shared resource setting according to a first operating requirement of a first core and a second operating requirement of a second core, the first core to

select an adjusted value for the independent resource setting that enables the first operating requirement to be satisfied.

4. The processor of claim 3, wherein the coordination logic is to translate the first operating requirement into a first target value, translate the second operating requirement into a second target value, compare the first operating requirement to the second operating requirement, select the second target value as the shared resource setting if the second operating requirement is greater than the first operating requirement and report a corrected operating requirement that results from the shared resource setting to the first core, the first core to select the adjusted value based on the corrected operating requirement.

5. The processor of claim 4, further including a performance state transition table, the coordination logic to use the performance state transition table to translate the first operating requirement into the first target value and translate the second operating requirement into the second target value.

6. The processor of claim 3, wherein the second core is to issue a modified operating requirement that corresponds to a reduced second target value, the coordination logic to adjust the shared resource setting based on the modified operating requirement, the first core to adjust the independent resource setting based on the adjusted shared resource setting.

7. The processor of claim 2, wherein the independent resource setting is to include a setting selected from a set including an independent clock throttle setting and an architecture throttle setting.

8. The processor of claim 2, wherein the coordination logic is to coordinate a plurality of shared resource settings with a plurality of independent resource settings.

9. The processor of claim 1, wherein the shared resource setting is to include a setting selected from a set including a voltage setting and a frequency setting.

10. The processor of claim 1, wherein the plurality of operating requirements is to include a plurality of power requirements.

11. The processor of claim 1, wherein the plurality of operating requirements is to include a plurality of performance requirements.

12. A method comprising:

issuing a plurality of operating requirements from a corresponding plurality of processor cores; and

coordinating a shared resource setting of the plurality of processor cores with the plurality of operating requirements.

13. The method of claim 12, further including coordinating the shared resource setting with an independent resource setting of at least one of the plurality of processor cores.

14. The method of claim 13, wherein coordinating the shared resource setting with the independent resource setting includes:

selecting the shared resource setting according to a first operating requirement of a first processor core and a second operating requirement of a second processor core; and

selecting an adjusted value for the independent resource setting that enables the first operating requirement to be satisfied.

15. The method of claim 14, wherein selecting the shared resource setting includes:

translating the first operating requirement into a first target value;

translating the second operating requirement into a second target value;

comparing the first operating requirement to the second operating requirement;

selecting the second target value as the shared resource setting if the second operating requirement is greater than the first operating requirement;

reporting a corrected operating requirement that results from the shared resource setting to the first core; and

selecting the adjusted value based on the corrected operating requirement.

16. The method of claim 15, wherein translating the first and second operating requirements includes using a performance state transition table.

17. The method of claim 14, further including:

issuing a modified operating requirement from the second processor core;

adjusting the shared resource setting based on the modified operating requirement; and

adjusting the independent resource setting based on the adjusted shared resource setting.

18. The method of claim 13, wherein the coordinating the shared resource setting with the independent resource setting includes coordinating the shared resource setting with a setting selected from a set including an independent clock throttle setting and an architecture throttle setting.

19. The method of claim 13, further including coordinating a plurality of shared resource settings with a plurality of independent resource settings.

20. The method of claim 12, wherein the coordinating includes coordinating at a setting selected from a set including a voltage setting and a frequency setting.

21. The method of claim 12, wherein the coordinating includes coordinating the shared resource setting with a plurality of power requirements.

22. The method of claim 12, wherein the coordinating includes coordinating the shared resource setting with a plurality of performance requirements.

23. A system comprising:

a clock source having a shared frequency setting; and

a processor coupled to the clock source, the processor having a plurality of cores to issue a corresponding plurality of operating requirements and coordination logic coupled to the plurality of cores, the coordination logic to coordinate a shared frequency setting of the plurality of cores with the plurality of operating requirements.

24. The system of claim 23, wherein the coordination logic is to coordinate the shared resource setting with an independent resource setting of at least one of the plurality of cores.

25. The system of claim 24, wherein the coordination logic is to select the shared frequency setting according to a first operating requirement of a first core and a second operating requirement of a second core, the first core to select an adjusted value for the independent resource setting that enables the first operating requirement to be satisfied.

26. The system of claim 25, wherein the coordination logic is to translate the first operating requirement into a first frequency, translate the second operating requirement into a second frequency, compare the first operating requirement to the second operating requirement, select the second frequency as the shared frequency setting if the second operating requirement is greater than the first operating requirement and report a corrected operating requirement that results from the shared resource setting to the first core, the first core to select the adjusted value based on the corrected operating requirement.

27. The system of claim 25, wherein the second core is to issue a modified operating requirement that corresponds to a reduced second frequency, the coordination logic to adjust the shared frequency setting based on the modified operating requirement, the first core to adjust the independent resource setting based on the adjusted shared frequency setting.

28. The system of claim 24, wherein the independent resource setting is to include a setting selected from a set including an independent clock throttle setting and an architecture throttle setting.

29. The system of claim 23, wherein the plurality of operating requirements is to include a plurality of power requirements.

30. The system of claim 23, wherein the plurality of operating requirements is to include a plurality of performance requirements.

31. A method comprising:

issuing a first operating requirement from a first processor core;

issuing a second operating requirement from a second processor core;

translating the first operating requirement into a first target value;

translating the second operating requirement into a second target value;

comparing the first operating requirement to the second operating requirement;

selecting the second target value as a shared resource setting if the second operating requirement is greater than the first operating requirement;

selecting an adjusted value for an independent resource setting that enables the first operating requirement to be satisfied;

issuing a modified operating requirement from the second processor core;

adjusting the shared resource setting based on the modified operating requirement; and

adjusting the independent resource setting based on the adjusted shared resource setting.

32. The method of claim 31, wherein the shared resource setting includes a setting selected from a set including a voltage setting and a frequency setting.

33. The method of claim 31, wherein the independent resource setting includes a setting selected from a set including an independent clock throttle setting and an architecture throttle setting.

* * * * *