

local variable data is stored in the upper portion causes state transition to state **708** (arrow **723**). This transition makes the upper portion the active portion so that the upper portion can be used to access the local variables stored therein.

[0084] In accordance with at least one embodiment of the invention, a state variable is maintained to indicate the state of the RAMset. For example, **FIG. 14** shows eight exemplary states and thus the state variable would have at least eight different values, each value corresponding to a different state. The algorithm discussed herein updates the state variable when the RAMset transitions from one state to another and, at least in part, uses the value of the RAMset state variable to determine the appropriate actions to be performed (e.g., II. Clean, R.SET(--), etc.) for each transition.

[0085] While the preferred embodiments of the present invention have been shown and described, modifications thereof can be made by one skilled in the art without departing from the spirit and teachings of the invention. The scope of protection is not limited by the description set out above. Each and every claim is incorporated into the specification as an embodiment of the present invention.

What is claimed is:

1. A processor adapted to couple to external memory, comprising:

a controller;

data storage operated by said controller, said data storage configurable to operate in either a cache policy mode in which a miss results in an access of the external memory or in a scratch pad policy mode in which a miss does not result in an access of the external memory;

wherein said data storage comprises a first portion and a second portion, and wherein only one of said portions is active at a time, the non-active portion being unusable; and

wherein, when the active portion does not have sufficient capacity for additional data to be stored therein, the other portion becomes the active portion.

2. The processor of claim 1 wherein, when said data storage has insufficient space to accept additional data, the controller cleans only one portion of the data storage, thereby copying valid contents of the data portion to the external memory.

3. The processor of claim 1 wherein said data storage changes from the scratch pad policy mode to the cache policy mode to cause data located in the external memory to be loaded back into the active portion on a cache miss.

4. The processor of claim 1 wherein said data storage changes from the cache policy mode to the scratch pad policy mode upon switching from one portion being the active portion to another portion being the active portion.

5. The processor of claim 1 wherein an oscillation occurs in which the two portions take turns being the active portion without needing to perform an access to external memory.

6. The processor of claim 1 wherein said data storage comprises cache memory.

7. The processor of claim 1 wherein said data storage is usable to store multiple sets of local variables, each set of local variables being associated with a specific executable software method.

8. The processor of claim 7 wherein, upon return from a first method to a second method and if the set of local variables associated with the second method are stored in the external memory, the controller causes the data storage to operate in the cache policy to thereby cause the set of local variables associated with the second method to be copied back into the active portion upon a miss that targets at least one local variable from among said set of local variables.

9. A system, comprising:

a communications transceiver;

a first memory;

a controller communicatively coupled to said communications transceiver and said memory; and

a second memory operated by said controller, said second memory configurable to operate in either a cache policy mode in which a miss results in an access of the first memory or in a scratch pad policy mode in which a miss does not result in an access of the first memory;

wherein said second memory comprises a first portion and a second portion, and wherein only one of said portions is active at a time, the non-active portion being unusable; and

wherein, when the active portion does not have sufficient capacity for additional data to be stored therein, the other portion becomes the active portion.

10. The system of claim 9 wherein, when said second memory has insufficient space to accept additional data, the controller cleans only one portion of the second memory, thereby copying valid contents of the data portion to the first memory.

11. The system of claim 9 wherein said second memory changes from the scratch pad policy mode to the cache policy mode to cause data located in the first memory to be loaded back into the active portion on a cache miss.

12. The system of claim 9 wherein said second memory changes from the cache policy mode to the scratch pad policy mode upon switching from one portion being the active portion to another portion being the active portion.

13. The system of claim 9 wherein an oscillation occurs in which the two portions take turns being the active portion without needing to perform an access to first memory.

14. The system of claim 9 wherein the system comprises a cellular telephone.

15. The system of claim 9 wherein said second memory comprises cache memory.

16. The system of claim 15 wherein the controller and second memory are included in a processor and the first memory comprises memory external to said processor.

17. The system of claim 9 wherein said second memory is usable to store multiple sets of local variables, each set of local variables being associated with a specific executable software method.

18. The system of claim 9 wherein, upon return from a first method to a second method and if the set of local variables associated with the second method are stored in the first memory, the controller causes the second memory to operate in the cache policy to thereby cause the set of local variables associated with the second method to be copied