RAMset space greater than the available space to be allocated in the RAMset. Consequently, a portion of the RAMset is cleaned (i.e. copied to external memory **106**) to make room for new data. This cleaning process is illustrated by arrow **705** which points to RAMset state **704**. In particular, the clean operation only cleans the upper portion of the RAMset. The data in the upper portion represents the oldest data in the RAMset and is copied to the corresponding page of external memory. The R.set(++) command is also performed at this time to allocate a new external memory page to the RAMset.

[**0073**] At state **704** the upper portion of the RAMset can again be used to store new local variables for newly invoked methods. The upper portion therefore becomes the active portion of the RAMset. At this point, the upper portion of the RAMset is the active portion, the lower portion of the RAMset contains valid data but is not currently used as the active portion, and the initial data in the upper portion from state **700** (or other data from states **706** or **710**) has been copied to external memory.

[**0074**] If insufficient space in the upper portion is available for the local variables of additional methods to be invoked, the lower portion of the RAMset can then be used for such additional local variables. In state **704**, however, the lower portion of the RAMset may already have valid local variables and thus a clean operation (II.CLEAN command) is performed to first clean the lower portion so that the lower portion can be used for additional local variables. This process is depicted via arrow **707** which points back to state **702**.

[**0075**] While at state **704** (also in SPP mode), new methods can be invoked and allocations of storage space in the upper portion of the RAMset can be performed for usage by such new methods. Of course, called methods may return back to their previous calling methods and eventually, the method that caused the first allocation of the upper portion at state **704** may return back to its calling method. That return is illustrated by arrow **711**, which points to RAMset state **706** (also in SPP mode). At RAMset state **706**, therefore, the lower portion of the RAMset again becomes the active portion. From the lower portion in state **706**, a method may be invoked which again exceeds the available capacity of the lower portion thereby causing the upper portion to become the active portion as identified by arrow **709** which transitions back to state **704**. Again, an oscillation can occur between states **704** and **706** (identified by oppositely pointing arrows **709** and **711** in dashed circle **691**), but such oscillations do not require any memory accesses and therefore can be performed with little time and little power consumption.

[**0076**] From state **706**, with the bottom portion being active, if a return is to be performed to a prior method whose local variables were stored in the upper portion of the RAMset and such data has been copied to external memory **106** (in a prior clean operation of the upper portion), the RAMset transitions to state **708** by way of return arrow **713**. Because the data associated with upper portion of the RAMset has been saved off to external memory, a flush of the upper portion is performed to invalidate the upper portion. Further, the upper portion of the RAMset, now the active is transitioned to the CP mode to permit the previously saved data to be loaded into the RAMset's upper portion.

[**0077**] From state **708**, if a return is performed to a prior method whose local variables are associated with the lower portion of the RAMset but have been saved off to external memory, the RAMset operates according to state **714** still in the CP mode (arrow **721**). A R.SET(--) command is performed to free the current memory page and restore the RAMset base address to the previous base address. Also, a flush of the bottom portion if performed to cause the bottom portion's data to be retrieved from external memory.

[**0078**] Going back to state **708**, if RAMset storage space is needed for a new method and the extra storage is not available in the currently active upper portion, the RAMset operates according to state **710**. In state **710**, the RAMset operates in the SPP mode and the bottom portion becomes the active portion for storing local variables. This invocation is illustrated by arrow **715**. A return to the method that caused the bottom portion to become active may be performed back to state **708** (arrow **717**). An oscillation between states **708** and **710**, designated by oppositely pointing arrows **715**, **717** within dashed circle **693** do not require any external memory accesses and therefore can be performed in relatively little time and with relatively little power consumption.

[**0079**] From state **710**, an invocation of a method that exceeds the storage capacity of the active lower portion takes the RAMset to a different state, in particular, state **704**. This transition is shown by way of arrow **719** and also requires a clean of the upper portion to be performed to save the data already present in the upper portion so that the upper portion of the RAMset can be used for additional local variables.

[**0080**] From state **700**, a return to a method whose local variables are associated with the lower portion but have been saved to external memory can be performed with the RAMset now operating to state **714**. This transition is identified by arrow **729** and a flush of the lower portion is performed along with a change in the allocation policy to the CP mode. The change to the CP mode causes previously cleaned data from external memory to be re-loaded into the corresponding lines of the lower portion of the RAMset.

[**0081**] An oscillation can also occur between states **712** and **714** between the lower and upper portions of the RAMset. The oscillations are indicated by oppositely pointing invocation arrow **725** and return arrow **727** within dashed circle **694**. This oscillation occurs without accesses to external memory and thus requires little time and power. As with the oscillation between states **708** and **710**, the oscillation between states **712** and **714** require a change in allocation policy as shown. RAMset state **712** is in the SPP mode because the needed local variable data is already in the upper portion. State **714** is in the CP mode because the needed data must be retrieved from external memory and re-loaded into the lower portion of the RAMset.

[**0082**] From state **712**, an invocation of a method that exceeds the storage capacity of the active upper portion takes the RAMset to a different state, and in particular, state **702**. This transition is shown by way of arrow **731** and also requires a clean of the lower portion to be performed to save the data already present in the lower portion so that the lower portion of the RAMset can be used for additional local variables.

[**0083**] Finally, from state **714** in which the lower portion is active, a method that returns to a calling method whose