CLAIMS:

1. A system comprising:

a radio to receive content;

a shared resource;

a host processor to execute a software module associated with the shared resource; and

a shader dispatch module including, an interface to receive a plurality of signals from the software module, wherein the plurality of signals are to include a first signal that indicates whether a draw call associated with the content accesses the shared resource, a second signal that indicates whether a boundary of the draw call has been reached, and a third signal that indicates whether the draw call has a coherency requirement, and

a draw call manager to selectively dispatch a workload corresponding to the draw call in a shader invocation based on the plurality of signals, wherein the selective dispatch includes issuing a first plurality of signals with a first draw call and a second plurality of signals with a second draw call, and the first plurality of signals and the second plurality of signals enable an internal tracking of the first draw call and the second draw call at a pipeline level of a graphics pipeline, wherein the graphics pipeline is to include a series of stages having a plurality of shader invocations, and wherein the second draw call is to progress through one or more stages of the graphics pipeline independent of an entirety of the graphics pipeline being flushed; and

a display to visually present the content.

2. The system of claim 1, wherein the draw call manager includes:

a global counter of pending globally observed draw calls; and

a coherency unit to withhold the workload from the shader invocation only if the first signal indicates that the draw call accesses the shared resource, the second signal indicates that a beginning of the draw call has been reached, and the third signal indicates that the draw call has a coherency requirement, wherein the workload is withheld until the global counter has reached zero.

3. The system of claim 2, wherein the draw call manager further includes:

an increment unit to increment the global counter in response to the second signal indicating that an end of the draw call has been reached; and

a decrement unit to decrement the global counter in response to an acknowledgement from a downstream graphics component.

4. The system of claim 1, wherein the draw call manager includes:

a group identifier first in first out (FIFO) buffer;

an allocator to allocate a group identifier for a group of threads if the first signal indicates that the draw call writes to the shared resource and the second signal indicates that a beginning of the draw call has been reached; and

a push unit to add the group identifier to the FIFO buffer.

5. The system of claim 1, wherein the draw call manager includes:

a thread counter;

an increment unit to increment the thread counter for each thread in a thread group associated with the draw call;

a launch unit to dispatch the threads in the thread group to the shader invocation; and

a decrement unit to decrement the thread counter in response to a retirement of each thread in the thread group.

6. The system of claim 1, wherein the draw call manager includes:

a group identifier first in first out (FIFO) buffer; and

a global observation unit to post a globally observed notification in response to a top entry in the FIFO buffer having a group identifier with a thread counter that is zero and a global counter being greater than zero, wherein the thread counter is to track each thread in a thread group associated with the draw call and the global counter is to track pending globally observed draw calls.

7. A method of operating a shader dispatch module, comprising:

receiving a plurality of signals from a software module associated with a shared resource, wherein the plurality of signals includes a first signal that indicates whether a draw call accesses the shared resource, a second signal that indicates whether a boundary of the draw call has been reached, and a third signal that indicates whether the draw call has a coherency requirement; and

selectively dispatching a workload corresponding to the draw call in a shader invocation based on the plurality of signals, wherein the selective dispatch includes issuing a first plurality of signals with a first draw call and a second plurality of signals with a second draw call, and the first plurality of signals and the second plurality of signals enable an internal tracking of the first draw call and the second draw call at a pipeline level of a graphics pipeline, wherein the graphics pipeline includes a series of stages having a plurality of shader invocations, and wherein the second draw call

progresses through one or more stages of the graphics pipeline independent of an entirety of the graphics pipeline being flushed.

8. The method of claim 7, wherein selectively dispatching the workload includes:

withholding the workload from the shader invocation only if the first signal indicates that the draw call accesses the shared resource, the second signal indicates that a beginning of the draw call has been reached, and the third signal indicates that the draw call has a coherency requirement, wherein the workload is withheld until a global counter of pending globally observed draw calls has reached zero.

9. The method of claim 8, further including:

incrementing the global counter in response to the second signal indicating that an end of the draw call has been reached; and

decrementing the global counter in response to an acknowledgement from a downstream graphics component.

10. The method of claim 7, further including:

allocating a group identifier for a group of threads if the first signal indicates that the draw call writes to the shared resource and the second signal indicates that a beginning of the draw call has been reached; and

adding the group identifier to a first in first out buffer.

11. The method of claim 7, further including:

incrementing a thread counter of each thread in a thread group associated with the draw call;

dispatching the threads in the thread group to the shader invocation; and

decrementing the thread counter in response to a retirement of each thread in the thread group.

12. The method of claim 7, further including:

posting a globally observed notification in response to a top entry in a first in first out buffer having a group identifier with a thread counter that is zero and a global counter being greater than zero, wherein the thread counter tracks each thread in a thread group associated with the draw call and the global counter tracks pending globally observed draw calls.

13. At least one non-transitory computer readable storage medium comprising a set of instructions which, when executed by a shader dispatch module of a computing device, cause the shader dispatch module to:

receive a plurality of signals from a software module associated with a shared resource, wherein the plurality of signals are to include a first signal that indicates whether a draw call accesses the shared resource, a second signal that indicates whether a boundary of the draw call has been reached, and a third signal that indicates whether the draw call has a coherency requirement; and

selectively dispatch a workload corresponding to the draw call in a shader invocation based on the plurality of signals, wherein the selective dispatch includes issuing a first plurality of signals with a first draw call and a second plurality of signals with a second draw call, and the first plurality of signals and the second plurality of signals enable an internal tracking of the first draw call and the second draw call at a pipeline level of a graphics pipeline, wherein the graphics pipeline is to include a series of stages having a plurality of shader invocations, and wherein the second draw call is to progress through one or more stages of the graphics pipeline independent of an entirety of the graphics pipeline being flushed.

14. The at least one non-transitory computer readable storage medium of claim 13, wherein the instructions, when executed, cause the shader dispatch module to:

withhold the workload from the shader invocation only if the first signal indicates that the draw call accesses the shared resource, the second signal indicates that a beginning of the draw call has been reached, and the third signal indicates that the draw call has a coherency requirement, wherein the workload is withheld until a global counter of pending globally observed draw calls has reached zero.

15. The at least one non-transitory computer readable storage medium of claim 14, wherein the instructions, when executed, cause the shader dispatch module to:

increment the global counter in response to the second signal indicating that an end of the draw call has been reached; and

decrement the global counter in response to an acknowledgement from a downstream graphics component.

16. The at least one non-transitory computer readable storage medium of claim 13, wherein the instructions, when executed, cause the shader dispatch module to:

allocate a group identifier for a group of threads if the first signal indicates that the draw call writes to the shared resource and the second signal indicates that a beginning of the draw call has been reached;

add the group identifier to a first in first out buffer.

17. The at least one non-transitory computer readable storage medium of claim 13, wherein the instructions, when executed, cause the shader dispatch module to:

increment a thread counter of each thread in a thread group associated with the draw call;

dispatch the threads in the thread group to the shader invocation; and

decrement the thread counter in response to a retirement of each thread in the thread group.

18. The at least one non-transitory computer readable storage medium of claim 13, wherein the instructions, when executed, cause the shader dispatch module to:

post a globally observed notification in response to a top entry in a first in first out buffer having a group identifier with a thread counter that is zero and a global counter being greater than zero, wherein the thread counter is to track each thread in a thread group associated with the draw call and the global counter is to track pending globally observed draw calls.

19. A shader dispatch module comprising:

an interface to receive a plurality of signals from a software module associated with a shared resource, wherein the plurality of signals are to include a first signal that indicates whether a draw call accesses the shared resource, a second signal that indicates whether a boundary of the draw call has been reached, and a third signal that indicates whether the draw call has a coherency requirement; and

a draw call manager to selectively dispatch a workload corresponding to the draw call in a shader invocation based on the plurality of signals, wherein the selective dispatch includes issuing a first plurality of signals with a first draw call and a second plurality of signals with a second draw call, and the first plurality of signals and the second plurality of signals enable an internal tracking of the first draw call and the second draw call at a pipeline level of a graphics pipeline, wherein the graphics pipeline is to include a series of stages having a plurality of shader invocations, and wherein the

second draw call is to progress through one or more stages of the graphics pipeline independent of an entirety of the graphics pipeline being flushed.

20. The shader dispatch module of claim 19, wherein the draw call manager includes:
a global counter of pending globally observed draw calls; and
a coherency unit to withhold the workload from the shader invocation only if the first signal indicates that the draw call accesses the shared resource, the second signal indicates that a beginning of the draw call has been reached, and the third signal indicates that the draw call has a coherency requirement, wherein the workload is withheld until the global counter has reached zero.

21. The shader dispatch module of claim 20, wherein the draw call manager further includes:
an increment unit to increment the global counter in response to the second signal indicating that an end of the draw call has been reached; and
a decrement unit to decrement the global counter in response to an acknowledgement from a downstream graphics component.

22. The shader dispatch module of claim 19, wherein the draw call manager includes:
a group identifier first in first out (FIFO) buffer;
an allocator to allocate a group identifier for a group of threads if the first signal indicates that the draw call writes to the shared resource and the second signal indicates that a beginning of the draw call has been reached; and
a push unit to add the group identifier to the FIFO buffer.

23. The shader dispatch module of claim 19, wherein the draw call manager includes:
a thread counter;
an increment unit to increment the thread counter for each thread in a thread group associated with the draw call;
a launch unit to dispatch the threads in the thread group to the shader invocation; and
a decrement unit to decrement the thread counter in response to a retirement of each thread in the thread group.

24. The shader dispatch module of claim 19, wherein the draw call manager includes:
a group identifier first in first out (FIFO) buffer; and
a global observation unit to post a globally observed notification in response to a top entry in the FIFO buffer having a group identifier with a thread counter that is zero and a global counter being greater than zero, wherein the thread counter is to track each

thread in a thread group associated with the draw call and the global counter is to track pending globally observed draw calls.