OpenMP* Directives and Clauses Summary

This topic provides a summary of the OpenMP* directives and clauses.

OpenMP Directives

Directive	Description
PARALLEL END PARALLEL	Defines a parallel region.
DO END DO	Identifies an iterative worksharing construct in which the iterations of the associated loop should be executed in parallel.
SECTIONS END SECTIONS	Identifies a non-iterative worksharing construct that specifies a set of structured blocks that are to be divided among threads in a team.
SECTION	Indicates that the associated structured block should be executed in parallel as part of the enclosing sections construct.
SINGLE END SINGLE	Identifies a construct that specifies that the associated structured block is executed by only one thread in the team.
PARALLEL FOR	A shortcut for a parallel region that contains a single for directive. **Note** The parallel or for OpenMP directive must be immediately followed by a for statement. If you place other statement or an OpenMP directive between the parallel or for directive and the for statement, the Intel C++ Compiler issues a syntax error.
FOR	Identifies an iterative work-sharing construct that specifies a region in which the iterations of the associated loop should be executed in parallel.
PARALLEL SECTIONS END PARALLEL SECTIONS	Provides a shortcut form for specifying a parallel region containing a single SECTIONS construct.

MASTER END MASTER	Identifies a construct that specifies a structured block that is executed by only the master thread of the team.
CRITICAL[*lock*] END CRITICAL[*lock*]	Identifies a construct that restricts execution of the associated structured block to a single thread at a time. Each thread waits at the beginning of the critical construct until no other thread is executing a critical construct with the same *lock* argument.
BARRIER	Synchronizes all the threads in a team. Each thread waits until all of the other threads in that team have reached this point.
ATOMIC	Ensures that a specific memory location is updated atomically, rather than exposing it to the possibility of multiple, simultaneously writing threads.
FLUSH [(*list*)]	Specifies a cross-thread sequence point at which the implementation is required to ensure that all the threads in a team have a consistent view of certain objects in memory. The optional *list* argument consists of a comma-separated list of variables to be flushed.
ORDERED END ORDERED	The structured block following an ORDERED directive is executed in the order in which iterations would be executed in a sequential loop.
THREADPRIVATE	Makes the named file-scope or namespace-scope variables specified private to a thread but file-scope visible within the thread.

OpenMP Clauses

Clause	Description
PRIVATE	Declares variables to be private to each thread in a team.
FIRSTPRIVATE	Provides a superset of the functionality provided by the private clause.
LASTPRIVATE	Provides a superset of the functionality provided by the private clause.
SHARED	Shares variables among all the threads in a team.
DEFAULT	Enables you to affect the data-scope attributes of variables.
REDUCTION	Performs a reduction on scalar variables.

ORDERED	The structured block following an ordered directive is executed in the order in which iterations would be executed in a sequential loop.
IF	If the if(scalar_logical_expression) clause is present, the enclosed code block is executed in parallel only if the scalar_logical_expression evaluates to TRUE. Otherwise the code block is serialized.
SCHEDULE	Specifies how iterations of the for loop are divided among the threads of the team.
COPYIN	Provides a mechanism to assign the same name to threadprivate variables for each thread in the team executing the parallel regio