OpenMP 5.0 API Reference Guide: Tasking

The OpenMP® API gives parallel programmers a simple and flexible interface for developing portable, scalable parallel applications in C/C++ and Fortran.

Functionality new/changed in OpenMP 5.0 is in this color, and in OpenMP 4.5 is in this color. [n.n.n] Sections in the 5.0 spec. Deprecation in the 5.0 spec.

Directives and Constructs

Team management constructs

parallel [2.4] [2.5]

Forms a team of threads and starts parallel execution.

```c
#pragma omp parallel [clause[ , ]clause ...] 
structed-block
```

private (list), firstprivate (list), shared (list), reduction (reduction-identifier: list)
proc_bind (master | close | spread)
allocate ((locator: list)
```c
#pragma omp parallel 
```

depobj construct

depobj [2.17.10.1]

Stand-alone directive that initializes, updates, or cancels an OpenMP depend object.

```c
#pragma omp depobj [depobj] clause
```

declare directive

declare reduction [2.19.5.7] [2.16]

Declares a reduction-identifier used in a reduction clause.

```c
#pragma omp declare reduction [reduction-identifier : type-name-list : combiner] [initializer-clause]
```

synchronization constructs

taskwait [2.17.5] [2.13.4]

Synchronizes on the completion of all tasks within the current parallel construct.

```c
#pragma omp taskwait [clause[ , ]clause ...] 
```

taskgroup [2.17.6] [2.13.5]

A taskgroup is created according to data-sharing attribute clauses on the taskgroup construct and any defaults that apply.

```c
#pragma omp taskgroup [clause[ , ]clause ...] 
```

Synchronization on task completion occurs on the last execution of the taskgroup construct.

```c
#pragma omp taskgroup 
```

task_reduction (reduction-identifier: list)
allocate ((locator: list)
```c
```

task [2.10.4] [2.11.2]

Synchronizes on the completion of all tasks within the current task.

```c
#pragma omp task [clause[ , ]clause ...] 
```

Synchronizes on the completion of all tasks within the current taskgroup.

```c
#pragma omp taskgroup [clause[ , ]clause ...] 
```

taskloop [2.10.1] [2.9.1]

Forms a team of threads and starts execution.

```c
#pragma omp taskloop [simd] 
```

num_teams (type-list)
grainsize (int-constant)
in_private (list)
```c
```

taskwait [2.18.2] [2.14.1]

Requests cancellation of the innermost enclosing region of the specified type.

```c
#pragma omp taskwait [clause[ , ]clause ...] 
```

taskyield [2.18.2] [2.13.4]

Introduces a user-defined cancellation point at which tasks can synchronize.

```c
#pragma omp taskyield 
```

cancellation point [2.18.2] [2.14.3]

Introduces a user-defined cancellation point at which tasks can synchronize.

```c
#pragma omp cancellation point [clause[ , ]clause ...] 
```

task_reduction (reduction-identifier: list)
allocate ((locator: list)
```c
```

taskgroup [2.17.6] [2.13.5]

Provides parallel execution on a team of threads.

```c
#pragma omp taskgroup [clause[ , ]clause ...] 
```

Synchronizes on the completion of all tasks within the current taskgroup.

```c
#pragma omp taskgroup 
```

Tablet event-handle: A type of event-handle.

depobj construct

depobj [2.17.10.1]

An OpenMP task construct that initializes, updates, or cancels an OpenMP depend object.

```c
#pragma omp depobj [depobj] clause
```

declare directive

declare reduction [2.19.5.7] [2.16]

An OpenMP task construct that declares a reduction-identifier used in a reduction clause.

```c
#pragma omp declare reduction [reduction-identifier : type-name-list : combiner] [initializer-clause]
```

A task construct that synchronizes on the completion of all tasks within the current task.

```c
#pragma omp task [clause[ , ]clause ...] 
```

A taskgroup construct that creates a taskgroup according to data-sharing attribute clauses on the taskgroup construct and any defaults that apply.

```c
#pragma omp taskgroup [clause[ , ]clause ...] 
```

A synchronization construct that synchronizes on the completion of all tasks within the current taskgroup.

```c
#pragma omp taskgroup 
```

A synchronization construct that synchronizes on the completion of all tasks within the current task.

```c
#pragma omp task [clause[ , ]clause ...] 
```

A synchronization construct that synchronizes on the completion of all tasks within the current parallel construct.

```c
#pragma omp taskwait [clause[ , ]clause ...] 
```

The OpenMP constructs feature simplicity and flexibility, allowing developers to easily implement parallel applications in C/C++ and Fortran.

C/C++ content

Fortran content
parallel master [2.13.6]
Shortcut for specifying a parallel construct containing one
master construct and no other statements.

|Pragma omp parallel master [clause, [clause]...]
|structured-block
|$omp parallel master [clause, [clause]...]
|$omp end parallel master

|clause: Any clause used for parallel or master.

master taskloop [simd] [2.13.7-8]
Shortcut for specifying a master construct containing a
taskloop [simd] construct and no other statements.

|Pragma omp master taskloop [simd] [clause, [clause]...]
|for-loops
|$omp master taskloop [simd] [clause, [clause]...]
|$omp end master taskloop [simd]

|clause: Any clause used for master or taskloop [simd].

parallel master taskloop [simd] [2.13.9-10]
Shortcut for specifying a parallel construct containing a
master taskloop [simd] construct and no other statements.

|Pragma omp parallel master taskloop [simd] [clause, [clause]...]
|for-loops
|$omp parallel master taskloop [simd] [clause, [clause]...]
|$omp end parallel master taskloop [simd]

|clause: Any clause used for parallel or master taskloop [simd] directives, except the in_reduction clause.

Clauses

All list items appearing in a clause must be visible according to the scoping rules of the base language. Not all of the clauses listed in this section are valid on all directives.

Allocate Clause [2.11.4]
allocate [allocorator, list]
Specify the memory allocator to be used to obtain storage for private variables of a directive.

allocate %lg
\* An integer expression of the omp_allocatort kind

Data Sharing Attribute Clauses [2.19.4] [2.15.3]
Applies only to variables whose names are visible in the construct on which the clause appears.

shared (l)
Declares list items to be shared by generated tasks.

private (l)
Declares list items to be private to a task or a SIMD lane.

firstprivate (l)
Declares list items to be private to a task, and initializes each of them with the value that the corresponding original item has when the construct is encountered.

lastprivate ([lastprivate-modifier] list)
Declares one or more list items to be private to an implicit task or SIMD lane, and causes the corresponding original list item to be updated after the end of the region.

lastprivate-modifier: conditional

Depend Clause [2.17.11] [2.13.9]
Enforces additional constraints on the scheduling of tasks or loop iterations, establishing dependences only between sibling tasks or between loop iterations.

depend [depend-modifier, dependence-type : locator-list]
dependence-modifier: iterator (iterators-definition)
dependence-type: in, out, inout, mutextinoutset, depobj

- in: The generated task will be dependent of all previously generated sibling tasks that reference at least one of the list items in an out or inout dependence-type list.
- out and inout: The generated task will be dependent of all previously generated sibling tasks that reference at least one of the list items in an in, out, or inout dependence-type list.
- mutextinoutset: If the storage location of at least one list item matches that of one appearing in a depend clause with in, out, or inout dependence-type on a construct from which a sibling task was previously generated, then the generated task will be a dependent task of that sibling. If the storage location of at least one of the list items is the same as that of a list item appearing in a depend clause with a mutextinoutset dependence-type on a construct from which a sibling task was previously generated, then the sibling tasks will be mutually exclusive.

Tasking Clauses [2.10] [2.9]
affinity [aff-modifier, locator-list]
A hint to execute closest to the location of the list items.
aff-modifier is iterator (iterators-definition).
allocate [allocorator, list]
See Allocate Clause, page 9 of this guide.

collapse (n)
Constant positive integer expression specifying how many loops are associated with the taskloop construct.

final [scalar-expression] [C/C++]
final [scalar-logical-expression] [C++]

- final: The generated task will be a final task if the expression evaluates to true.

Grain Size [2.13.9-10]
grain-size [grain-size]
C/C++
Causes the number of logical loop iterations assigned to each created task to be <= the minimum of the value of the grain-size expression and the number of logical loop iterations, but less than two times the value of the grain-size expression.

if [(task : scalar-expression) C/C++]
if [(task : scalar-logical-expression) C++]
For See If Clause, page 10 of this guide.

in_reduction [reduction-identifier : list]
See Reduction Clauses in this guide.

mergable
Specifies that the generated task is a mergeable task.

nogroup
Prevents an implicit taskgroup region to be created.

num_tasks [num-tasks]
Create as many tasks as the minimum of the num-tasks expression and the number of logical loop iterations.

priority [priority-value]
A non-negative numerical scalar expression that specifies a hint for the priority of the generated task.

reduction [default, reduction-identifier : list]
See Reduction Clauses in this guide.

untied
If present, any thread in the team can resume the task region after a suspension.

Iterators

iterator [2.1.6]
Identifiers that expand to multiple values in the clause on which they appear.

iterator (iterators-definition)
iterators-definition:
iterator-specifier, ..., iterator-specifier

iterator-specifier: [iterator-type] identifier + range-specification

- iterator-type: A type name or specifier.
- identifier: A base language identifier.
- range-specification: begin : end; step
  - begin, end: Expressions for which their types can be converted to iterator-type
  - step: An integral expression.

Copyright © 2018 OpenMP Architecture Review Board.
Permission to copy without fee or part of this material is granted, provided the OpenMP Architecture Review Board copyright notice and the title of this document appear.
Notice is given that copying is by permission of the OpenMP Architecture Review Board. Portions of this product / publication may have been derived from the OpenMP Language Application Program Interface Specification.