OpenMP Implementation-Defined Behaviors for OMPi v3.0.0

This document enlists behaviors of the OMPi compiler, which are described as implementation-defined in the OpenMP specifications.

1 Internal Control Variables (ICVs)

Specific ICVs and their initial values are given in the table below. Please notice that OMPi can utilize multiple and different threading libraries, each of which may have its own initial / default values for some of these ICVs; library-specific ICVs are marked by an asterisk (*). Values in the table below refer to the default threading library (pthreads).

ICV name	Initial value
affinity-format-var	`"Level: %L, TID: %n/%N, Affinity: %A"`
bind-var	`false` (*)
def-allocator-var	*not yet implemented*
default-device-var	1 if at least 1 device attached, 0 (host) if none attached
dyn-var	`true` (*)
max-active-levels-var	no limit (*)
num-procs-var	*not yet implemented*
nthreads-var	# available cores (*)
place-partition-var	`cores` if `HWLOC` is available, `threads` ('processors') if not.
run-sched-var	`auto`
stacksize-var	OS default value (*)
thread-limit-var	no limit (*)
wait-policy-var	`active` (*)

(*) threading-library specific.

2 Dynamic Adjustment of Threads

When the dynamic adjustment of threads is disabled and the threading library cannot provide the requested number of threads, the application is aborted with an advice to enable dynamic adjustment either programmatically or using the corresponding environmental variable.

3 Worksharing Loop Directive

- If the runtime schedule has been selected and the *run-sched-var* ICV is set to `auto`, then the method for distributing the iterations is selected according to the environment variable `OMPI_SCHED_AUTO`.
- If the auto schedule has been selected and `OMPI_SCHED_AUTO` has not been set or does not conform to the specified format, then the iterations are distributed using a `static` schedule.
- For a collapsed loop, the variable used to compute the iteration count is of type `unsigned long int`.

4 Sections Construct

Scheduling of the structured `section` blocks among threads is competitive; threads are assigned `section` blocks on a first-come first-serve basis.

5 Single Construct

The selection of a thread to execute a structured `single` block is competitive; the first thread to ask for it, gets to execute it.

6 Distribute construct

If no `dist_schedule` clause is specified then the iterations are distributed using a `static` schedule. For a collapsed loop, the variable used to compute the iteration count is of type `unsigned long int`.

7 Processor

A 'processor' is whatever the OS system calls report.

8 Affinity

For the `close`/`spread` affinity policies, if $T > P$ and P does not divide T evenly, places/subpartitions $0, 1, \ldots, (T \bmod P) - 1$ will contain $S + 1$ threads, while the rest will contain S threads, where $S = \lfloor P/T \rfloor$.

9 Device-specific behaviors

Devices are the host and any additional attached compute units. The host is always the device with id 0. In this version, OMPi can support the Adapteva Epiphany accelerator as a device.

9.1 The `is_device_ptr` clause

OMPi only supports pointers returned by the `omp_target_alloc()` call and pointers specified through the `use_device_ptr` clause.

9.2 Declare target procedures

For every procedure within a `declare target` directive, the same version is generated for all currently supported devices.

10 Runtime Routines

- `omp_set_num_threads()` : if the argument is not a positive integer, then it is assumed to be equal to 1.
- `omp_set_schedule()` : there are four additional loop schedules defined. Trapezoid Self Scheduling, Taper Method, Fixed Size Chunking, Factoring and Factoring2 can be selected with the values 1234 to 1237. The new methods require some additional inputs given with this method. Consult OMPi documentation for further details.
- `omp_get_schedule()` : there is no guarantee that the value returned by the second argumnet has any significant meaning for methods other than `static`, `dynamic` and `guided`.
- `omp_set_max_active_levels()` : if the argument is not a positive integer, then the routine simply returns. If it is called from within a explicit parallel region, the binding thread set is all threads.
- `omp_set_affinity_format()` : if it is called from within an explicit parallel region, the binding thread set is all threads and the final value of the affinity format is random.
- `omp_get_affinity_format()` : if it is called from within an explicit parallel region, the binding thread set is all threads.
- `omp_display_affinity()` , `omp_capture_affinity()` : if the format argument does not conform to the specified format, then the value of the affinity-format-var ICV is used.
- `omp_get_place_proc_ids()` : the numerical identifiers returned correspond to the unique ID of each 'processor'. Their order matches the one in which they are found in the place list of the execution environment.
- `omp_get_initial_device()` : returns 0, which is the id of the host device.
- `omp_set_default_device()` : if the value is a negative integer, or it is greater than the number of available devices minus one, the host becomes the default device.
- `omp_init_lock_with_hint()` : the hint is currently ignored; the call is equivalent to `omp_init_lock()`.
- `omp_init_nest_lock_with_hint()` : the hint is currently ignored; the call is equivalent to `omp_init_nest_lock()`.
- `omp_target_memcpy_rect()` : there is no limit on the number of dimensions.

11 Environmental Variables

- `OMP_SCHEDULE` : if the value of the variable does not conform to the specified format, it is considered to be equal to `auto`.
- `OMP_NUM_THREADS` : if any list value is not a positive integer, the default number of threads is used (which is threading-library specific; see the table above). It the requested number of threads exceeds the capabilities of the threading library (which cannot happen when using the default threading library) the program is aborted with an informative message.
- `OMP_PROC_BIND` : if the value of the variable is not `true`, `false` or a comma-seperated list of `primary`, `master`, `close` or `spread`, then it is considered to be `false`. If an initial thread cannot be bound to the first place of the place list, binding for it and it's child threads is disabled. If the value is `true` the thread affinity policy is by default `close`.

- `OMP_PLACES` : if defined using an explicit list of places described by non-negative numbers, these numbers correspond to the unique ID of each 'processor'. If the exclusion operator ! is encountered while parsing the e.v., all previous occurrences of the place following the operator will be removed from the place list of the execution environment. No action is taken in case a numeric value cannot be mapped to a 'processor' (unavailable or not).

  The abstract names recognized are `threads`, `cores`, `sockets`, `numa_domains` and `ll_caches`. If the Portable Hardware Locality (`HWLOC`) software package is available topology detection is conducted in order for the abstract names to be mapped into 'processor' IDs. In case `HWLOC` is not available, the default place list is used which consists of as many places as the number of 'processor's available in the underlying system and each place contains a single 'processor' ID.

  If the place list is created by appending the number n to an abstract name, then a) if n is greater than the available resources, all the resources are used and b) if n is smaller than the available resources, the first n resources are used.

- `OMP_DYNAMIC, OMP_NESTED, OMP_CANCELLATION` : if the value of the variable is neither `true` nor `false`, then it is considered to be `false`.

- `OMP_STACKSIZE` : if the value of the variable does not conform to the specified format, it is considered to be equal to 256KB.

- `OMP_WAIT_POLICY` : if the value of the variable is neither active nor passive, it is ignored. The details of the wait policy behavior are threading-library specific. For the default threading library, the wait policy is always active irrespectively of the value of this environmental variable.

- `OMP_MAX_ACTIVE_LEVELS` : if the value is not a positive integer, it is ignored. The default behavior, for the case where the requested number of levels is larger than what can be supported, is threading-library specific (does not concern the default threading library). **Conflicting value of** `OMP_NESTED`: if `OMP_NESTED` is `false`, no nested parallelism level is created, even if `OMP_MAX_ACTIVE_LEVELS` > 1.

- `OMP_THREAD_LIMIT` : if the value is not a positive integer, it is ignored. The default behavior, for the case where the requested number of threads is larger than what can be supported, is threading-library specific (does not concern the default threading library).

- `OMP_DEFAULT_DEVICE` : if the value is a negative integer, or it is greater than the number of available devices minus one, the host becomes the default device.