Goal: To support the following GEOPM markup routines through ‘GEOPM’ service in Caliper.

The following table lists GEOPM markup routines, description and the corresponding attributes in Caliper to be reported to GEOPM by the proposed GEOPM service in Caliper.

|  |  |  |
| --- | --- | --- |
| **GEOPM markup** | **Description** | **Attributes/Callbacks** |
| int geopm\_prof\_region(  const char \**region\_name*,  long *policy\_hint*,  uint64\_t \**region\_id*) | Registers an application region. The *region\_name* and *policy\_hint* parameters are input parameters, and the *region\_id* is output.  (Note: GEOPM doest not support nested regions as of now, so only return the top-level region ID) | Caliper: region ID  Unresolved: policy hint  Scope: process local  Callback: pre\_begin\_evt |
| int geopm\_prof\_enter(  uint64\_t *region\_id*)  int geopm\_prof\_exit(  uint64\_t *region\_id*) | Called by the compute application to mark begin and end of the profiled compute region associated with the *region\_id*. | Caliper: region ID  Scope: process local  Callback: pre\_begin\_evt, pre\_end\_evt |
| int geopm\_prof\_progress(  uint64\_t *region\_id*,  double *fraction*) | Called by compute application in single threaded context to signal the fractional progress, *fraction* through the work required to complete the region where *fraction* is between 0 and 1. | Caliper: region ID  Scope: process local-only  Callback: pre\_end\_evt  Additional call required to report: ‘fraction’ |
| int geopm\_prof\_epoch(  *void)* | Called once for each pass through a computational loop containing inter-node synchronization events. Acts as a beacon signal emitted by each MPI rank as it begins a loop iteration. | Attribute:  Scope: process local  Callback: pre\_end\_evt  This is mapped to a the end of region ID named ‘mainloop’ to indicate end of the main loop as required by GEOPM. |
| int geopm\_prof\_disable(  const char \**feature\_name*) | Called at application start up to disable a profiling feature. | Scope: process local  Not implemented in GEOPM. |
| int geopm\_tprof\_create(  int *num\_thread*,  size\_t *num\_iter*,  size\_t *chunk\_size*,  struct geopm\_tprof\_c \*\**tprof*)  int geopm\_tprof\_destroy(  struct geopm\_tprof\_c \**tprof*) | Create and release a thread profiling object, *tprof*, which extends the functionality of the profiling interface to report progress within threaded regions.  Assume a fixed number of threads, *num\_thread*, which are performing work sharing on a list of tasks *num\_iter* long (e.g. an OMP parallel for loop with *num\_iter* loops). | Attribute: Annotation, loop  Scope: thread local  Callback: pre\_set\_evt |
| int geopm\_tprof\_increment(  struct geopm\_tprof\_c \**tprof*,  uint64\_t *region\_id*,  int *thread\_idx*) | Called after a thread has completed each work unit to report progress. | Attribute: Annotation, loop  Scope: thread local-only  Callback: pre\_end\_evt  Must not be called along with geopm\_prof\_progress( ) |