
sc_time object: data type for specifying time

e.g. sc_time waitTime(5, sc_us)

sc_time_resolution: sets global time resolution (default is 1 picosecond)

sc_time_stamp(): returns sc_time object for current simulation time

sc_start(): starts simulation with optional specification of simulation time

e.g. sc_start() -> start simulation until it is stopped explicitly

e.g. sc_start(10, sc_ns); -> start simulation and stop after 10 ms of simulation time

sc_step(): steps simulation

wait(0): can be used to wait for fixed length of time

e.g. wait(10, sc_ms)

wait(waitTime);
Initialization: All simulation processes are added within simulation engine with designation of process state:

Runnable: Process can be executing during current simulation step
Running: Process is the current process running
Waiting: Process is waiting (either for fixed time or some "event")

Evaluate: Evaluate phase of simulation is used to run all runnable processes

- Each process will run until either a return (for `sc_method`) or `wait` (for `sc_thread`) is reached
- If `wait(sc_time)` is used, the process and time will be added to scheduling priority queue
- Order of processes to run is unspecified

Advance Time: Once all processes have been run (i.e., there are no runnable processes), the simulation time will advance to closest time specified within scheduling queue

```
sc_start() -> Init -> Evaluate -> Advance Time
```

- `sc_step()` or "nothing to do" i.e., no runnable or waiting processes
System Events: Event occurs at specific instance of time and has no duration

- Only has two basic operations

  1. wait(); waits for occurrence of event
     e.g. wait(someEvent)

  2. notify(); triggers occurrence of event

     # Causes any process waiting for event to be immediately
     # moved from wait state to runnable state

     # Can also have delayed events that scheduled in the future
     e.g. someEvent, notify(10, SC_NS),

     # Causes notify for event to be delayed by specified time

SPECIAL CASE: Zero-Time Delayed Notification

notify(SC_ZERO_TIME);
notify(0, SC_SEC);

# Causes notify for event to be delayed until all current
# runnable processes are evaluated

# Implies that any process waiting for event will be
# moved from wait to runnable state after all current
# runnable states are evaluated

NOTE: The distinction between immediate and delayed notification is
Update: Update stage used to determine if processes need to be run (i.e. moved from wait to runnable) due to zero-time delayed notifications.

sc_signal: used to model channels in which channels can be concurrently updated.

read(): Returns current value of signal.

write(): Specifies next value of signal that is updated with a zero-time delay.

Any process waiting for event on signal (e.g., sc_signal.event() or sc_METHOD sensitive to event) will become runnable after all current runnable processes are evaluated.

Updating signal values occurs within update stage and not evaluate.

⇒ Provides concurrency semantics. All updates to signals in current evaluation phase will be updated at same simulation time (one delta cycle later).