

## **Reducing Design Simulation Time by Efficiently Controlling Multiple Threads**

The design simulation time can be reduced drastically by efficiently controlling the sequence of simulation on Specman.

“e” allows you to define methods that execute within a single point of simulation time (within zero time) or methods that execute over multiple cycles. The first type of method is referred to as a regular method. The second type is called a Time Control Methods (TCM).

Within a single “e” program, multiple TCMs can execute either concurrently or in sequence, along parallel but separate threads. A TCM can also have internal branches, which are multiple action blocks executing concurrently. Some of the functions performed by test bench like reset, back door initialization, initialization of BFM and monitors can be controlled by TCM.

Every Specman verification environment should have at least one TCM that should start running as soon as the simulation starts. Often, one TCM is the main (or root) initiator of many TCMs running in parallel. The unit that contains the main TCM should be the top-level unit.

The following code shows some examples of TCMs:

```
extend enet_agent {
  --Main TCM for agent unit.
  main() @sys.any is {
    start tcm_agent();
    method_agent();
  };
  run() is also {
    start main();
  };
};
```

```
extend enet_bfm {
  --Main TCM for bfm unit.
  main() @sys.any is {
    start tcm_bfm();
    method_bfm();
  };
  run() is also {
    start main();
  };
};
```

Starting only main TCM from the run also (i.e single control over all the TCM threads) helps delay the simulation.

Here is a partial list of actions that agent may perform at start of the operation while the user may want to delay the actions:

- Check initial signal values.
- Measure clock period.
- Reset.
- Initialize. (do some actions that are not sending data items. e.g. auto negotiation)
- Start its internal units: BFM, collector, monitor, drivers
- Backdoor initialization. (do the init stuff in zero time)
- Start the stop mechanism

The user can delay or sequence the simulation of agent and BFM by adding the following code:

```
extend enet_bfm {  
    event start_sim;  
    main() @clk is first {
```

```
        -- Wait for the start_sim before starting the simulation.  
        -- User can emit the start_sim event after an appropriate delay.  
        sync @start_sim  
    };  
};
```

```
extend enet_bfm {  
    --Binding the bfm's start_sim event to agent's start_sim event.  
    event start_sim is only cycle@parent_agent.start_sim;  
    main() @clk is first {  
        -- Wait for the start_sim before starting simulation of the bfm unit.  
        sync @start_sim;  
    };  
};
```

#### **Conclusion:**

Thus we see that TCMs can be used for efficiently sequencing the simulation of different functions and also reduce the design simulation time.