sc-during: Parallel Programming with SystemC for Loosely Timed Models

**TLM design-flow**

<table>
<thead>
<tr>
<th>Traditional Design-Flow</th>
<th>Transaction-Level Model based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specification, Algorithm</td>
<td>Specification, Algorithm</td>
</tr>
<tr>
<td>RTL Design</td>
<td>RTL Design</td>
</tr>
<tr>
<td>Synthesis</td>
<td>Synthesis</td>
</tr>
<tr>
<td>Factory</td>
<td>Factory</td>
</tr>
<tr>
<td>Software Development</td>
<td>Software Development</td>
</tr>
<tr>
<td>Integration</td>
<td>Integration</td>
</tr>
<tr>
<td>Validation</td>
<td>Validation</td>
</tr>
</tbody>
</table>

**Example**

- CPU
- ITC
- VGA
- Timer

- TLM Bus
- Data RAM
- Instruction RAM
- GPIO

**Uses of TLM**

- Early software development
- Hardware verification
- Architecture exploration

**Contents of a model**

- What is needed for software
- ... and only that

**Simulated time**

- Wall-clock time
- Simulated time
- Time elapsed
- Computation

**Simulation Parallelism**

**Simulation speed**

- Pure RTL
- RTL + cosimulation
- TLM
- HW emulation

<table>
<thead>
<tr>
<th>Number of CPU in the platform</th>
<th>thread, during_quant</th>
<th>thread, during_sync</th>
<th>ondemand, during_quant</th>
<th>ondemand, during_sync</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speedup</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Processes in SystemC and sc-during**

- sc-during

**Results**

**Tasks with Duration**

- Process A:
  ```
  f();
  // Computation
  wait(20);
  // Time taken by f
  ```

- Process P:
  ```
  g();
  wait(20);
  during(15, h);
  ```

**Sketch of implementation**

```c
void during(sc_core::sc_time duration, std::function<void()> routine) {
  std::thread t(routine); // create thread
  sc_core::wait(duration); // let SystemC execute
  t.join(); // wait for thread completion
}
```

- during(5, f);
- create thread
- wait(d)
- join thread

**Try it!**

- Open Source (< 1500 LOC)
- http://sc-during.forge.imag.fr/