DUAL-TOP FRAMEWORK FOR TESTBENCH ACCELERATION

1. Employ two separated HVL and HDL sides
2. Model all timed testbench code for synthesis on the HDL side, leaving the HVL side untimed
3. Devise a transaction-level, function-based communication interface between HVL and HDL sides

Accelerate UVM Testbench
Single Unified Testbench for Simulation and Acceleration

TOOL FRAMEWORK FOR TESTBench ACCELERATION

- TOP FRAMEWORK
- SINGLE UNIFIED
- Tool

TESTBENCH ACCELERATION PERFORMANCE DEMYSTIFIED

\[ \tau_{\text{total}} = \tau_{\text{HVL}} + \tau_{\text{HVL-HDL}} + \tau_{\text{HDL}} \]

- Host Workstation
- HVL-HDL Channels
- HVL side (SV)
- HVL side (SV Connect)
- HDL side (SV)
- HDL side (SV Connect)
- UVM Agent
- UVM Proxy

Throughput: \( \frac{\tau_{\text{HVL}}}{\tau_{\text{HVL-HDL}} + \tau_{\text{HDL}}} \)

H/W Bound: \( \frac{\tau_{\text{HVL}}}{\tau_{\text{HVL-HDL}} + \tau_{\text{HDL}}} \times \left(1 + \frac{\tau_{\text{HVL-HDL}}}{\tau_{\text{HDL}}}\right) \)

TACKLING TESTBENCH ACCELERATION PERFORMANCE

S/W Time \( \tau_{\text{HVL}} \)
- Workstation executing HVL-side testbench threads causing emulated design clocks to stop
- Profile conventional simulation runs to analyze the testbench portion – discount the RTL DUT to be allotted to the emulator
- Code SV/UVM testbench for simulation performance, not just functionality
- Maximize constraint solver performance
- Move assertions and coverage models to the HDL-side where applicable and capacity is not of concern

S/W-H/W Communication Time \( \tau_{\text{HVL-HDL}} \)
- Workstation-emulator context-switching and data transfer
- Profile acceleration runs
- Choose an efficient HVL-HDL communication scheme appropriate for the application at hand
  - For reactive applications (with instantaneous transfer of control and data), use VIF-based inbound and outbound functions and tasks, or equivalently, DPI-C export and import functions
  - For streaming interfaces (with producer and consumer decoupled), use SCMDI transaction pipes (e.g. audio, video, Ethernet, etc.)
- Maximize H/W-S/W concurrency

H/W Time \( \tau_{\text{HDL}} \)
- Emulator executing HDL-side RTL DUT along with BFM and clock/reset generators
- Optimize for best possible emulator clock frequency, which is a function of the combinational logic critical path
- Typical optimization considerations to achieve higher emulator frequency and level out capacity are critical path analysis, clock utilization (inactive edges, edge alignment), interfaces parallelism, etc.
- Use automated performance and capacity advisor technology

SV/UVM Testbench Acceleration Case Studies

<table>
<thead>
<tr>
<th>Component</th>
<th>Application Processor</th>
<th>Network Switch</th>
<th>Graphics Sub-System</th>
<th>Mobile Display Processor</th>
<th>Memory Controller</th>
<th>Face Recognition Engine</th>
<th>Wireless Multi-Media Sub-System</th>
<th>Hard Engine Controller I</th>
<th>Hard Engine Controller II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>32GB</td>
<td>100Gbps</td>
<td>4GB</td>
<td>4GB</td>
<td>3GB</td>
<td>3GB</td>
<td>3GB</td>
<td>3GB</td>
<td>3GB</td>
</tr>
<tr>
<td>Frequency</td>
<td>4GHz</td>
<td>20Gbps</td>
<td>1GHz</td>
<td>1GHz</td>
<td>1GHz</td>
<td>1GHz</td>
<td>1GHz</td>
<td>1GHz</td>
<td>1GHz</td>
</tr>
</tbody>
</table>