IP-XACT based SoC Interconnect Verification Automation

YoungRae Cho, YoungSik Kim, Seonil Brian Choi
Samsung Electronics co. ltd
AGENDA

• SOC INTERCONNECT VERIFICATION
• CHALLENGES IN SOC INTERCONNECT VERIFICATION
• IP-XACT BASED AUTOMATION
• RESULT
• UVM-based environment
• VIPs for the interfaces
  • Covers the address space and routes
• Protocol validation
• Interconnect Validator VIP
  • Monitor and check transactions within a interconnect fabric
- Spreadsheet-based VIP configuration for hundreds of interfaces

<table>
<thead>
<tr>
<th>Interface_type</th>
<th>Interface_mode</th>
<th>id_width</th>
<th>address_width</th>
<th>date_width</th>
<th>lower_address</th>
<th>upper_address</th>
<th>agent_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AXI4 MASTER</td>
<td>14</td>
<td>64</td>
<td>64 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AXI4 MASTER</td>
<td>14</td>
<td>64</td>
<td>64 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AXI4 MASTER</td>
<td>14</td>
<td>64</td>
<td>64 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>4</td>
<td>32</td>
<td>128 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>9</td>
<td>32</td>
<td>128 0x13f0_0000 0x13f2_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>1</td>
<td>1</td>
<td>64 0x1600_0000 0x1c07_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>32</td>
<td>32</td>
<td>32 0x1692_0000 0x1692_ffff</td>
<td>PASSIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APB3 MASTER</td>
<td>32</td>
<td>32</td>
<td>32 0x1691_0000 0x1691_ffff</td>
<td>PASSIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APB3 MASTER</td>
<td>32</td>
<td>32</td>
<td>32 0x1690_0000 0x1690_ffff</td>
<td>PASSIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Master / Slave mapping table

<table>
<thead>
<tr>
<th>Interface_type</th>
<th>Interface_mode</th>
<th>id_width</th>
<th>address_width</th>
<th>date_width</th>
<th>lower_address</th>
<th>upper_address</th>
<th>agent_type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AXI4 MASTER</td>
<td>14</td>
<td>64</td>
<td>64 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AXI4 MASTER</td>
<td>14</td>
<td>64</td>
<td>64 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AXI4 MASTER</td>
<td>14</td>
<td>64</td>
<td>64 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>4</td>
<td>32</td>
<td>128 0x2000_0000 0x0ff_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>9</td>
<td>32</td>
<td>128 0x13f0_0000 0x13f2_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>1</td>
<td>1</td>
<td>64 0x1600_0000 0x1c07_ffff</td>
<td>ACTIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AHB MASTER</td>
<td>32</td>
<td>32</td>
<td>32 0x1692_0000 0x1692_ffff</td>
<td>PASSIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APB3 MASTER</td>
<td>32</td>
<td>32</td>
<td>32 0x1691_0000 0x1691_ffff</td>
<td>PASSIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>APB3 MASTER</td>
<td>32</td>
<td>32</td>
<td>32 0x1690_0000 0x1690_ffff</td>
<td>PASSIVE</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CHALLENGES IN SOC IC VERIFICATION

- Instance hierarchy and map between the VIP and DUT's signals: tens of thousands of lines

```vhdl
'define XOUT_17_VPATH tbench_top.dut.BLK_AUD.AUD/Core.CTRL
'define XOUT_18_VPATH tbench_top.dut.BLK_GPS.GPS/Core.PU
'define HOUT_00_VPATH tbench_top.dut.BLK_CAM.CAM/Core.CU

force axi_ica_xout_18_if0_arid = 'XOUT_18_VPATH.i_AXIS2_ARID;
force axi_ica_xout_18_if0_arvalid = 'XOUT_18_VPATH.i_AXIS2_ARVALID;
force axi_ica_xout_18_if0_arready = axi_ica_xout_18_if0_arready;
force axi_ica_xout_18_if0_araddr = 'XOUT_18_VPATH.i_AXIS2_ARADDR + 32'h0408_0000;
force axi_ica_xout_18_if0_arlen = 'XOUT_18_VPATH.i_AXIS2_ARLEN;
force axi_ica_xout_18_if0_arsize = 'XOUT_18_VPATH.i_AXIS2_ARSIZE;
force axi_ica_xout_18_if0_arburst = 'XOUT_18_VPATH.i_AXIS2_ARBURST;
force axi_ica_xout_18_if0_arprot = 'XOUT_18_VPATH.i_AXIS2_ARPROT;
force axi_ica_xout_18_if0_arlock = 'XOUT_18_VPATH.i_AXIS2_ARLOCK;
force axi_ica_xout_18_if0_arcache = 'XOUT_18_VPATH.i_AXIS2_ARCACHE;
```

- SoC spec is frequently changed
IP-XACT FOR DESIGN FLOW

- **IP-XACT (IEEE 1685)** is an XML format that defines design meta data which enable automated configuration and integration through tools.
- Design spec information in IP-XACT:
  - Ports, interfaces mapping, instance hierarchy and so on.
- For Interconnect verification, all interconnect components should have routing information:
  - In design integration flow, the routing information is not needed.
  - It should be added for verification.

**Interface definition**
- AXI
- AHB
- APB

**IP Packaging**
- SFRs

**SoC/Sub-system**
• The routing information in IP-XACT is complicated.
• It is quite a burden to IP designers.
IP-XACT ROUTING INFORMATION

- The script converts the text-based routing information into IP-XACT
IP-XACT ROUTING INFORMATION

• Routing information integration flow for AMBA bridges

Address Range = 0x0-0x3fff

• Routing information integration flow for one to one bridges

Address Range = 0x0-0xffff_ffff
IP-XACT ROUTING INFORMATION

- 90% routing information is added by automated script
- GUI editor for IP-XACT routing information

1. Create memoryMap for slave input interface
2. Create addressSpace for output master interfaces with range
3. Drag and map addressSpace to memoryMap with based address
IP-XACT FOR IC VERIFICATION

• Port mapping, bit-with, hierarchical architecture and configurations
• Routing and memory map from any master to any slave can be calculated
• Interconnect verification environment can be generated using this information
IP-XACT BASED AUTOMATION

• GUI of IP-XACT based interconnect verification environment generation
IP-XACT BASED AUTOMATION Flow

- Port mapping, bit-with, hierarchical architecture and configurations
- Routing information for all interconnect components.
- Load IP-XACT DB
- Choose master & slave for interconnect verification or load saved configuration
- Generate Interconnect verification environment
- VIPs for the interfaces
- Interconnect Validator VIP
RESULT

• 50 master and 400 slave interfaces in our last SoC project
  • Two weeks to create -> one day
  • Only for end-to-end interconnect -> Any hierarchical level of SoC
  • Multiple test benches for subsystem or SoC
• Reduced time to test bench modification
  • Generate new test bench for every RTL update at an early stage of the project
  • Less false-negative, time to run more tests, finding more bugs in the early stage.
SUMMARY

• Verifying interconnect is a significant challenge
  • Interface VIP, interconnect validator VIP and excel based automated flow
• Configuration to comply DUT speciation change is also difficult
  • The number of masters and slaves are enormous
  • DUT speciation is frequently changed
• Design metadata (IP-XACT) for Interconnect verification
  • Routing information for all interconnect component
  • Most routing information can be packaged using automated method
• Interconnect verification environment is generated based on the IP-XACT
  • TB setup time: two weeks -> one day
  • Less false-negative, time to run more tests, finding more bugs in the early stage
THANK YOU!