Creating an Accelerator

Zynq
Vivado HLS 2014.2 Version

After completing this module, you will be able to:

- Describe embedded system development flow in Zynq using Vivado
- List the steps involved in creating an hardware accelerator
- State how an accelerator created in Vivado HLS is used in Vivado Design Suite

Objectives
Creating an Accelerator 26- 3

**Outline**

- Creating IP-XACT Adapters
- Integrating the IP-XACT Adapter in AXI System
- Summary
### Adapter Cores

**List of Adapter Cores for reference**

<table>
<thead>
<tr>
<th>Type</th>
<th>Core Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AXI4</td>
<td>AXI4M</td>
<td>AXI4 Master interface</td>
</tr>
<tr>
<td></td>
<td>AXI4Stream</td>
<td>AXI4 stream interface</td>
</tr>
<tr>
<td></td>
<td>AXI4LiteS</td>
<td>AXI4Lite slave interface</td>
</tr>
</tbody>
</table>

### Using RESOURCE for external connections

**When the array is an argument of the top-level function**
- The array/RAM is “off-chip”
- The type of RESOURCE determines the top-level IO ports

```c
void foo_top(int A[3*N], int x)
{
    int i;
    for (i = 0; i < N; i++)
}
```

**Add adapters to the RTL and create a bus interface**
- The principle is the same
- We define a resource for the port
  - Plus there are a few other options we’ll cover
Adding Adapters 1 ) Check the RTL & C IO

The first step in adding an adapter
- Make sure you have a suitable port on the RTL
- Example,
  - If you want an AXI Stream interface → must have an ap_fifo on the RTL
  - If an ap_fifo on the RTL is needed → pass pointer variable, an array, or a reference variable (in, out, but not in/out)

2) Assign an adapter Core to each Port

Once the I/O adapter is known
- Add the core as a resource on the RTL port
- The cores are listed in the Vivado HLS Library Guide
  - And available from the menu in the GUI
3) Optionally group and rename buses

**Metadata bundle buses**
- Allow buses to be grouped with new names
- Multiple resources can have the same name
  - Will be grouped into the same bus
- In this example, port “x” will be
  - AXI4 Slave interface
  - And renamed fir_io
  - Any other adapter named fir_io will be grouped with this port

**Port Name Mappings**
- Optionally, port may be name mapped in the port map field
  - {{output} {output_hs_vld, output_ap_ack}}
- The RTL port names are renamed on the adapter interface
  - In the above example, it will be renamed as output

---

4) Generate the Adapter

**Select Solution > Export RTL**
**Select IP Catalog, System Generator for Vivado or PCore for EDK**
**Click on Configuration… if you want to change the version number or other information**
- Default is v1_00_a
**Click on OK**
- The directory (ip) will be generated under the impl folder under the current project directory and current solution
- Only RTL code in Verilog will be generated, even if you select VHDL language
Generated impl Directory

- impl
- ip
- verilog
- vhdl

- drivers
- example
- misc
- subcore
- srlr
- verilog
- Makefile
- xfr_h.txt
- xfr_v.txt
- xfr_s.txt

- verilog
- extraction.tcl
- fr_c.rnd.dat
- fr_shift.reg.v
- xfr.v
- xfr.xc
- impl.bat
- projectspr
- fir.v
- voidpool
- voidpool
- voidpool
- voidpool

- Header file for slave interfaces
- Point IP Catalog to point to the ip directory
- IP Integrator will use this file
- Generated Verilog RTL Files

Outline

- Creating IP-XACT Adapters
- Integrating the IP-XACT Adapter in AXI System
- Summary
Integrate the IP-XACT Adapter in AXI System

- Create a new Vivado project, or open an existing project
- Set IP setting to point to the exported IP using Project Settings
- Invoke IP Integrator
- Construct (modify) the hardware portion of the embedded design by adding the IP-XACT adapter created in Vivado HLS
- Create (Update) top level HDL wrapper
- Synthesize and implement in Vivado
- Export the hardware description, and launch SDK

Integrate the IP-XACT Adapter in AXI System

- Set software repository to point to the created IP if needed
- Create a new [or update an existing] software board support package to include the driver for the IP
- Create an application project in the SDK
- Compile the software with the GNU cross-compiler in SDK
- Download the programmable logic’s completed bitstream
- Use SDK to download the program (the ELF file)
Creating IP-XACT Adapters

Integrating the IP-XACT Adapter in AXI System

Summary

Vivado HLS provides wide support of AXI interfaces, System Generator design, and Pcore for EDK
- Assign as an external resource, just like a RAM
- The choice of adapter is a function of the C variable type (pointer, etc.)

Start with the correct C argument type
- Verify the design at the C level
- Accept the default block-level I/O protocol
- Select the port-level I/O protocol that gives the required pcore adapter interface
- Specify the port to have the appropriate adapter RESOURCE
- Optionally group and rename ports

Export the design

Add the IP in Vivado’s IP Integrator