POLITECNICO DI MILANO

Target Customization and Tool Integration

Tutorial @ FPT Conference 2017 – Melbourne– Australia

Marco Lattuada

Politecnico di Milano
Dipartimento di Elettronica, Informazione e Bioingegneria
marco.lattuada@polimi.it

Outline

- Target Selection
- Integration with Simulator and Synthesis tools

Target Selection

- Target = FPGA device(+synthesis tool) + clock period
  - Different delays of FPGA elements (i.e., delay of a DSP)
  - Different sizes of FPGA elements (i.e., size of LUTs)
  - Different HDL description of memory elements
- Target device + target clock period can be specified
  - Default is
    - Target device: xc7z020-1clg484-VVD
    - Target clock period: **10ns**

Target Information

- Target information is embedded in XML files
  - Supported devices → included in bambu executable
  - New devices → must be passed to the tool
- XML file mainly contains characterization of functional units
  - Area
  - Delay
- XML files are automatically generated by means of eucalyptus (distributed in PandA)
  - New devices can be easily added

Currently supported devices

- Intel
  - Cyclone II: EP2C70F896C6, EP2C70F896C6-R
  - Cyclone V: 5CSEMA5F31C6
  - Stratix IV: EP4SGX530KH40C2
  - Stratix V: 5SGXEA7N2F45C1
- Lattice
  - ECP3: LFE335EA8FN484C
- Xilinx
  - Virtex 4: xc4vlx100-10ff1513
  - Virtex 5: xc5vlx110t-1ff1136 xc5vlx330t-2ff1738 xc5vlx50-3ff1153
  - Virtex 6: xc6vlx240t-1ff1156
  - Artix 7: xc7a100t-1csg324-VVD
  - Virtex 7: xc7vx330t-1ffg1157 xc7vx485t-2ffg1761-VVD xc7vx690t-3ffg1930-VVD
  - Zynq: xc7z020-1clg484-VVD (default), xc7z020-1clg484, xc7z020-1clg484-YOSYS-VVD

Command line options


```
--device-name=<value>
```

Specify the name of the device (see previous slide)

Default is xc7z020-1clg484 (Xilinx Zynq)

```
--clock-period=<value>
```

Specify the period of the clock signal (in nanoseconds)

Default is 10

Example:

```
--device-name=5SGXEA7N2F45C1 --clock-period=5
```

Interfacing Synthesis Tools

- Bambu can directly interface synthesis tools:
  - Quartus / Quartus Prime
  - ISE
  - Vivado
  - Diamond
- Bambu
  - generates synthesis scripts
  - collects information about generated solutions

```
--evaluation
```

Interfacing Synthesis Tools

- User can provide
  - VHDL/Verilog implementation of custom module
  - Constraint files
- Customization of design flow
  - User can provide XML files containing custom TCL scripts

Interfacing Simulation Tools

- Bambu can directly interface simulation tools
- Support to single tools must be enabled during configuration
  - isim
  - xsim
  - modelsim
  - icarus
  - verilator (enabled in VM)
- Bambu can
  - generate testbench + simulation scripts
  - collect data

Testbench Generation

- Testbench is automatically generated in Verilog by bambu starting from:
  - Randomly generated values
  - XML file `--generate-tb=<file.xml>`
  - command line option `--generate-tb=<values>`
  - Annotated C file
    - Support to `open`, `read`, `write` of files
- Result of the simulation is compared with
  - C input or
  - C generated from bambu IR
- Maximum allowed ULP can be set

First example– Write C Interface

- Synthesize a module which takes as input an array of integer with arbitrary size and returns the minimum and maximum value
  - Do not use structure

Solution Hint

Example: Code returning the maximum of array of 10 elements

```
void max(int input[10], int * max)
{
   int local_max = input[0];
   int i = 0;
   for(i = 0; i < 10; i++)
   {
      if(input[i] > local_max)
      {
         local_max = input[i];
      }
   }
   *max = local_max;
}
```

Possible Solution


```
void min_max(int * input, int num_elements, int * max, int * min)
{
   int local_max = input[0];
   int local_min = input[0];
   int i = 0;
   for(i = 0; i < num_elements; i++)
   {
      if(input[i] > local_max)
      {
         local_max = input[i];
      }
      else if(input[i] < local_min)
      {
         local_min = input[i];
      }
   }
   *min = local_min;
   *max = local_max;
}
```

Second example – Write testbench

- Write testbench for the module designed in the previous activity
  - Test arrays with different elements and different sizes

Solution Hint


```
--generate-tb=<xml_file>
```

```
<?xml version="1.0"?>
<function>
   <testbench input="0,1,2,3,4" num_elements="5"/>
</function>
```

Possible Solution


```
<?xml version="1.0"?>
<function>
   <testbench input="0,1,2,3,4" num_elements="5"/>
   <testbench input="0,1,2,3,4,5,6,7,8,9" num_elements="10"/>
   <testbench input="0,0,0,0,0,0,0,0,0,0" num_elements="10"/>
   <testbench input="0" num_elements="1"/>
</function>
```

Third example – Select target device and clock target period

- Compare the number of cycles required by a function executing 64 bit multiplication on the following target
  - xc4vlx100-10ff1513 – 66MHz
  - 5SGXEA7N2F45C1 – 200MHz
  - xc7vx690t-3ffg1930-VVD – 100MHz
  - xc7vx690t-3ffg1930-VVD – 333MHz
  - xc7vx690t-3ffg1930-VVD – 400MHz

Solution Hint


```
bambu module.c --device-name=xc4vlx100-10ff1513
--clock-period=15 --simulate
```

Solution


```
bambu module.c --device-name=xc4vlx100-10ff1513
--clock-period=15 --simulate

bambu module.c --device-name=5SGXEA7N2F45C1 --clock-period=5
--simulate

bambu module.c --device-name=xc7vx690t-3ffg1930-VVD
--clock-period=10 --simulate

bambu module.c --device-name=xc7vx690t-3ffg1930-VVD
--clock-period=3.3 --simulate

bambu module.c --device-name=xc7vx690t-3ffg1930-VVD
--clock-period=2.5 --simulate
```