EclipseCon Europe, Ludwigsburg
2017-10-24

Taming complex chip designs with beautiful diagrams
Titouan Vervack

redefine.digital.design: Helping you deal with complexity in VHDL and Verilog.
Ghent
What do we do?

Eclipse-based VHDL & SystemVerilog IDE
Chip design
Chip design
**Hardware Description Language (HDL)**

```vhdl
entity and_or_top is
  Port(
    A : in bit;  -- NAND gate input
    B : in bit;  -- NAND gate input
    Q : out bit  -- NAND gate output
  );
end and_or_top;

architecture Behavioral of and_or_top is
begin
  Q <= A nand B;  -- 2 input NAND gate
end Behavioral;
```

- **VHDL**
- **(System)Verilog**
Graphical editors with HDL

HDL code
Graphical editors with HDL

Graphical

Generate

HDL code
Graphical editors with HDL

Graphical → Generate → HDL code → Compile → Warnings & Errors
Graphical editors with HDL

Graphical

Generate

HDL code

Compile

Warnings & Errors
Graphical editors with HDL

Graphical → Generate → HDL code → Compile → Warnings & Errors
Graphical editors with HDL
Graphical editors with HDL
Graphical editors with HDL

Graphical ⇒ Generate ⇒ HDL code ⇒ Compile ⇒ Warnings & Errors
Graphical editors with HDL

Graphical

Generate

HDL code

Compile

Warnings & Errors

?
Graphical editors with HDL

Graphical view

Navigate

Generate

HDL code

Compile

Warnings & Errors
Graphical View of structure

```vhdl
entity foobar is
  port(a, b : in integer;
       c : out integer);
end foobar;

architecture RTL of foobar is
  signal sa, sb, sc, sd : integer;
begin
  sd <= a;
  block1 : entity foobar
    port map(a => sd, b => b, c => sa);
  block2 : entity foobar
    port map(a => sa, b => sa, c => sb);
  block3 : entity foobar
    port map(a => sa, b => sb, c => sc);
  block4 : entity foobar
    port map(a => sc, b => sc, c => c);
end RTL;
```
Text-based design with graphical views

ELK (Eclipse Layout Kernel)

KlighD
(Kieler Light Weight Diagrams)

JavaFx

Graphviz DOT

GEF & Zest
Graphical <-> source interaction

```vhdl
entity foobar is
  port(a, b : in integer;
  c       : out integer);
end foobar;

architecture RTL of foobar is
  signal sa, sb, sc, sd : integer;
begin
  sd <= a;
  block1 : entity foobar
    port map(a => sd, b => b, c => sa);
  block2 : entity foobar
    port map(a => sa, b => sa, c => sb);
  block3 : entity foobar
    port map(a => sa, b => sb, c => sc);
  block4 : entity foobar
    port map(a => sc, b => sc, c => c);
end RTL;
```
Graphical <-> source interaction
Graphical <-> source interaction

declaration of entity 'foobar'

architecture 'RTL' of 'foobar' is

begin

// Block Diagram

Entity declaration and architecture

// Code snippet

entity foobar is
  port(a, b : in integer;
       c : out integer);
end foobar;

architecture RTL of foobar is
  signal sa, sb, sc, sd : integer;
begin
  sd <= a;
  block1 : entity foobar
    port map(a <= sd, b => b, c => sa);
  block2 : entity foobar
    port map(a => sa, b => sb, c => sc);
  block3 : entity foobar
    port map(a => sc, b => sd, c => sb);
  block4 : entity foobar
    port map(a <= sc, b => sc, c => c);
end RTL;
Graphical <-> source interaction
Graphical <-> source interaction

```vhdl
entity foobar is
  port(a, b : in integer;
       c : out integer);
end foobar;

architecture RTL of foobar is
  signal sa, sb, sc, sd : integer;
begin
  sd <= a;
  block1 : entity foobar
    port map(a => sd, b => c, c => sa);
  block2 : entity foobar
    port map(a => sa, b => c, c => sb);
  block3 : entity foobar
    port map(a => c, b => sc, c => c);
  end RTL;
```
Text-based design with graphical views

Perfect solution?

- Cluttered
- Too much detail (e.g. 20MB SVG file)
- No focus
Text-based design with graphical views
Text-based design with graphical views
UI configuration

Results from Global War

- Shrinking of available design space
- Overly confusing interface
- A desire to go back to a simpler design

Font Color

- Automatic

Formatting of selected text

- Bulleted
- Select All
- New Style...

Pick formatting to apply

- Normal
- Clear Formatting
- Bulleted

Show: Available formatting
Goals:

- Filtering
- Mergable
- Reproducible
- Serializable
- Simplify collaboration
- Ability to highlight important parts
Our updated view

HDL code
Our updated view

HDL code

Navigate

Compile

Warnings & Errors
Our updated view

Graphical

Navigate

Generate

HDL code

Compile

Navigate

Warnings & Errors
Our updated view

Graphical

HDL code

Warnings & Errors

Compile

Navigate

Generate

DSL

Navigate

Compile

Our updated view
Our updated view

Graphical → HDL code → DSL → Warnings & Errors

Navigate

Generate

Compile
Our updated view

Graphical

Navigate

Generate

HDL code

Xtext

Compile

Warnings & Errors

DSL

Generate

Compile

Navigate
Separate file:

- Does not taint the original source code
- Multiple views for one source
  - Clocking/reset structure
  - Data flow
- Can be generated (group all wires between blocks)
DEMO
- DSL is fully covered

- UI: assume framework works

- SVG export: font size differences are a challenge
Testing

- DSL is fully covered \textit{Xtext}

- UI: assume framework works

- SVG export: font size differences are a challenge
Testing

- DSL is fully covered Xtext
- UI: assume framework works
- SVG export: font size differences are a challenge
Conclusion

- Use graphical views, not graphical editors
- One on one projection is often not what you want
- ELK + Xtext + Eclipse
Evaluate the Sessions
Sign in and vote at eclipsecon.org