vhd2vl

Larry Doolittle (LBNL)

April 17, 2023

OSDA/DATE, Antwerp

Disclaimer:

It's important to recognize that for me, this is a hobby project.

I am personally affiliated with LBNL, but vhd2vl is not.

I'm paid to do science and engineering with tools,

not develop general-purpose tools like this.

vhd2vl has never been funded by or distributed by LBNL.

vhd2vl

- what is it?
- where did it come from?
- how does it fit in the OSDA toolkit?
- what does it tell us about code?

vhd2vl converts (some dialects of synthesizable) VHDL to Verilog

- `https://github.com/ldoolitt/vhd2vl`
- licensed under GPLv2

“Can I convert VHDL code to Verilog” is a recurring if not frequently-asked question in developer forums

Typical answers:

- Why would you want to do that?
- No.
- Try some proprietary program.
- Try synthesizing with a proprietary program, then re-emitting as Verilog.
- Any results you get will be unreadable/unmaintainable.

History

“Can I convert VHDL code to Verilog”

- You can (or at least used to be able to) find tables showing 1:1 mapping of syntax between VHDL and Verilog, so applying the rules by hand was possible if tedious.

But then in Aug 2003 Vincenzo Liguori posted his answer to computer-programming-forum.com:

> I wrote a translator that supports a limited but useful subset of synthesisable VHDL. Although limited, this program correctly translated our Triple DES and JPEG cores. Since we have no commercial interest in such sw I decided to release it in the public domain under GPL.

- First written by Vincenzo Liguori of Ocean Logic Pty Ltd (in Australia), 2003
- Maintained by Larry Doolittle (in California) since 2005
- Maintained by Rodrigo Alejandro Melo (in Argentina) since 2017

History

“Can I convert VHDL code to Verilog” now that we have vhd2vl:

- Forum posters still give their stock answers
- VHDL written by pragmatic hardware engineers usually works great
- VHDL written by abstraction-enamored software people often crashes and burns
- vhd2vl may be inaccessible to non-*nix developers (anyone tried it in WSL?)

Another program with the same intent is vhdl2verilog, written in Java (1.6.x)

- `https://www.edautils.com/vhdl2verilog.html`

Apparently free to download, but not open source.

There is also a recipe to use GHDL for this purpose. Sounds great! I haven’t made it work.
I guess it will suffer from the unreadability-of-output problem.

The Icarus Verilog team made an attempt at this task, but that is abandoned and marked as deprecated.

- It can go the other way, converting Verilog to VHDL!

Example input and output – note some comments are preserved!

```
entity ifchain2 is port(
  clk, rstn : in std_logic;
  enable: in std_logic;
  result: out std_logic
);
end ifchain2;

architecture rtl of ifchain2 is
  signal counter : unsigned(3 downto 0);
  constant CLK_DIV_VAL : unsigned(3 downto 0) := to_unsigned(11,4);
begin

clk_src : process(clk, rstn) is
begin
     if (rstn = ’0’) then
          counter <= (others => ’0’);
          result <= ’0’;
     elsif (rising_edge(clk)) then -- Divide by 2 by default
          if (enable = ’1’) then
               if (counter = 0) then
                    counter <= CLK_DIV_VAL;
                    result <= ’1’;
               else
                    counter <= counter - 1;
                    result <= ’0’;
               end if; -- counter
          end if; -- enable
     end if; -- clk, rst_n
end process clk_src;
assert (counter < CLK_DIV_VAL) report "test case" severity error;
end rtl;
```

```
module ifchain2(
input wire clk,
input wire rstn,
input wire enable,
output reg result
);

reg [3:0] counter;
parameter CLK_DIV_VAL = 11;
  always @(posedge clk, posedge rstn) begin
    if((rstn == 1’b0)) begin
      counter <= {4{1’b0}};
      result <= 1’b0;
    end else begin
      // Divide by 2 by default
      if((enable == 1’b1)) begin
        if((counter == 0)) begin
          counter <= CLK_DIV_VAL;
          result <= 1’b1;
        end
        else begin
          counter <= counter - 1;
          result <= 1’b0;
        end
        // counter
      end
      // enable
    end
  end

endmodule
```

Unfair comparison

tool	Origin	Lines of code	regression tests	Contributors
vhd2vl	2003	2917	22	6 (2 current)
Verilator	2006?	135671	2095	200+
Yosys	2013	179609	>546	238
Icarus	1998	153457	>3142	84

Internally vhd2vl uses flex/bison heavily, but 1/3 of it is actually C99.

What is a program?

- Data is code! (buffer-exploit writers, and modern web browsers)
- Code is speech! (1995, Bernstein v. United States)
- Code is data! (every non-siloed developer starting with Grace Hopper)

What is a program?

Programs that manipulate other programs are the leverage needed to make programming a long-term investment.

Tools range from gcc to sed to bison to vhd2vl

- useless unless people can picture what they do, and apply them productively!

Important that a single program file can get processed by multiple tools

- grep for instances of an identifier
- linters
- translators (e.g., vhd2vl)
- simulators
- synthesizers
- formal verification
- editors and version control

Despite the recent hype, it's not obvious that A.I. means anything in this regime

- non-reproducible, not regression-testable

Conclusions

or the end of the beginning

vhd2vl can play a role in putting code to work in new ways

- Of course I'm always eager to hear about successes or get bug reports

Dank u wel – Merci
Thank you – Danke

for your attention and support