

Introduction to HDL

Hardware Description Language (HDL) is used to describe hardware (usually digital) circuits. It allows multiple level of circuit abstraction, thus a synthesis tool is usually required to actually figure out what a piece of code really represents on hardware. With a decent synthesis tool, it can be a good alternative to schematic drawing as a design entry method. Why is this good? Keeping track of clearly documented text files for a design is much better compared to library-based schematic file with

specific format (plus you get to use the cool SCMs like git ). Of course, a really good synthesis tool is necessary to create an accurate hardware design netlist.

The two most popular language in this category are **VHDL** (**V**ery **H**igh **S**peed **I**ntegrated **C**ircuit **H**DL) and **Verilog** HDL. Both languages are being used extensively although the trend seems towards Verilog (<https://www.opencores.org> and [Icarus](https://www.lcarus.com)). Personally, I'd prefer VHDL for it's completeness and strong type-casting feature. And no, I'm not saying that VHDL is better than Verilog... I was just stating my *personal* preference. I have used both languages and that's how I feel. But, I do think that Verilog has its merits like the fact that it has much closer link to simulation and simpler syntax. Saying that, I have to admit that I have been using Verilog A LOT for my recent work. Bottom line, choose one that you're comfortable with. But, if you're working for a small/medium-sized company which are not rich enough to buy support for both languages, you don't get to choose. Use whatever is given to you, or get another job. So, the REALLY bottom (under-bottom?) line, get to know both. Should I go on

with the other not-so-known languages? 

VHDL Background

1981	Initiated by US DoD to address hardware life-cycle crisis
1983-85	Development of baseline language by Intermetrics, IBM and TI
1986	All rights transferred to IEEE
1987	Publication of IEEE Standard
1987	Mil Std 454 requires comprehensive VHDL descriptions to be delivered with ASICs
1994	Revised standard (named VHDL 1076-1993)
2000	Revised standard (named VHDL 1076 2000, Edition)
2002	Revised standard (named VHDL 1076-2002)
2007	VHDL Procedural Language Application Interface standard (VHDL 1076c-2007)
2009	Revised Standard (named VHDL 1076-2008)

Verilog Background

1995	Verilog HDL became IEEE Std. 1364-1995
2001	- revised IEEE Std. 1364-2001
2005	- minor revision

```
extension => SystemVerilog
2005      - IEEE standard 1800-2005
```

Describing Hardware

Coding for Synthesis

As mentioned earlier, a good synthesis tool is required to translate HDL codes into hardware design netlist. What's a netlist? Well, it's a list of nets (a.k.a. interconnections) between circuit elements. It's exactly what you draw when you create a schematic but, in list form (conveniently in a text file). So, one way to make life easier for the synthesis tool (which is merely a computer program that is not exactly a genius ;P), we can write the HDL code in certain way so that it is clear to the tool what we actually want in the design. This is called coding for synthesis. And this topic, on its own, has created a lot of discussions among HDL users.

From:

<https://azman.my1matrix.cc/> - **Azman @MY1**

Permanent link:

https://azman.my1matrix.cc/doku.php?id=learn:hdl_coding&rev=1777342661

Last update: **2026/04/28 02:17**

