Questions to assess where You are.

Go over the items and check with Yourself, how wide is Your ASIC/FPGA world? People who work years in the same job/company can become oblivious to the world beyond. It can backfire when fired or looking for a new job.

Linux

1. Linux: can You use it? to what level?
2. Basic linux commands: alias grep fgrep sed ls chmod cd find
3. Text editors in Linux: what are Your preferred ones?
4. MarkDown documents: (.md extension) proficiency? using MD editor (like Typora) ?
5. Installation of linux tools (especially VLSI CAD ones) ?
6. Linux zsh, bash, tcsh or none?
7. Useful variables (setenv-s) : LD_LIBRARY_PATH, PATH, display, ... setenv vs set
8. Scripting languages (tcsh, bash, python, perl,...) and what were the applications
9. Version control tools: open source, commercials like SOS.
10. Do You have linux on Your laptop?

GIT

Are You: Active user, Familiar, Over-the-shoulder

1. git basics
    2. git tricks
        3. git frustrations
        4. git wrapped around with company specific wraps.

Open Source Tools

Not enough designers are aware, did You meet these?

1. iverilog (icarus)
2. verilator (more advanced)
3. graphviz (dot)
4. gtkwave / dinotrace
5. yosys

6. openSTA
7. openRoad
8. draw.io
9. wavedrom
10. register file generator

Commercial tools

You: Active use, Used long time ago, Familiar, Over-the-shoulder

1. Verilog simulator: Synopsys/Cadence/Mentor/Aldec
2. Synthesis: Synopsys/Cadence
3. static timing
4. linters
5. FPGA: Altera/Xilinx/?
6. TCL scripting for these tools?
7. How deep these tools buried in EDA environments?
8. Formal verification: Rule checkers.
9. Version control

Backends

11. logic equivalence
12. floor planning
13. place and route: placement, clock tree, routing.

Verilog

I can do these: Can You? Did You?

1. write control module (State machines and such).
2. write arithmetic module.
3. write rom like module.
4. integration of IPs.
5. integration of whole chip, including IOs..
6. configuration and generation of IP from cadence, synopsys or others.
7. Generated code by script. If yes, examples.

8. if You put code into open source(GitHub) - examples.
9. Downloaded verilog modules from open source.
10. build test bench and used it to debug modules.
11. did verification with SV-UVm or E-Verisity
12. Line Coverage and functional coverage
13. GateLevel simulation with or without SDF.
14. Mixed level simulation. if yes, What setup?
15. difference Synthesizable RTL vs TestBench (non-synthesizable) verilog
16. FPGA based verification or Palladium.
17. Post silicon validation and bringup.

Backend topics

Partial list of backend activities (How familiar are You with the below?)

1. DFT why? DFT ideas in general
2. ATPG for scans
3. BIST for RAMs
4. Compressed Scan. At speed Scan.
5. LVS : logic versus schematic.
6. DRC : design rules check (as in layout).
7. Antenna : antenna rules check (as in layout).

My toolbox

Tools i like to use that i maintain.

1. regfile.py: register file generator
2. zdraw.py: schematic editor
3. genver.py : python based macro pre-processor of verilog
4. pyver.py : verilog code manipulator.
5. vcd_python3 : intelligent scan of VCD files
6. python-driven-verification: python-simulator connection through VPI interface.

Popular comm standards

(Good to know stuff)

1. AXI4
2. APB
3. AHB
4. UART
5. SPI
6. I2C
7. JTAG
8. RGMI (and relatives)
9. MIPI (whole family)
10. DDR (PHY and Controller)
11. USB (many layers: Phy, Controller, upper level)
12. PCIE (many layers)

Less popular

1. CAN-FD CAN-XL
2. LIN
3. SPMI

Encryption / Signatures

- AES
- tripple DES
- XTEA