UC San Diego
Electrical and Computer Engineering
JACOBS SCHOOL OF ENGINEERING

Lecture 1: Introduction

UCSD ECE 111

Prof. Farinaz Koushanfar

Fall 2024

Some slides are courtesy of Prof. Lin Prof. Karna, or Prof. Turakia's ECE111 courses

ECE 111 - Fall 2023 course info

- **Class section/meeting times**
  - Section ID: 552096
  - Lectures: Tues/Thurs 14:00-15:20PM; Location: WLH 2204
  - In-Person Discussions: Wed 4-4:50PM (starting Oct 9); Location: CNTRE 214
  - Virtual Location: Meeting ID: 948 6397 0932; Passcode 004453
  - https://ucsd.zoom.us/j/94863970932?pwd=iXcQXbLYjOaAQTdOJlrmglCYMSyeir.1
- **Instructor:** Prof. Farinaz Koushanfar
  - Office hours: Tues 12-14pm (and by email appt)
  - (in person and by email appointment)
  - Office: Jacobs Hall 4201 (FAH 4th floor MICS Collab)
- **Teaching Assistants:**
  - Devanshi Panchal <depanchal@ucsd.edu>
  - Arnav Kulkarni <a3kulkarni@ucsd.edu>
  - Virtual office hours: 14-15:50pm MWF, and by email appointment

ECE 111 - Fall 2023 course info

- **Course webpage**
  - https://canvas.ucsd.edu/

- **All announcements will be done through Piazza. We will add a Piazza section on Canvas so you can sign up there**

- **Grading**
- 35% homeworks (5 out of 6 best / 7% each)
- 5% Participation
- 20% Final Project part 1
- 30% Final Project part 2
- 10% Final Presentation

Teaching Platform and Resources

- Discussion Forum: Piazza
  - Live streaming of lectures using Podcast
  - Lectures will also be recorded, and links will be published to students on canvas
  - Each week lecture online meetings are scheduled and published on class canvas

- Canvas will be used to publish course material and resources:
  - Lecture slides, Zoom meetings, homework, project, quiz, tools instructions, learning resources

- Piazza for Q&A and Announcements:
  - All announcements on piazza such as quiz date, polls, project discussion sessions and more
  - Using piazza students can ask any questions on lectures, homework, projects, quiz and more

- Gradescope to publish and upload homeworks:
  - Homework's and final project will be published on gradescope which is linked inside canvas
  - Students to upload their homework and final project report on gradescope through canvas

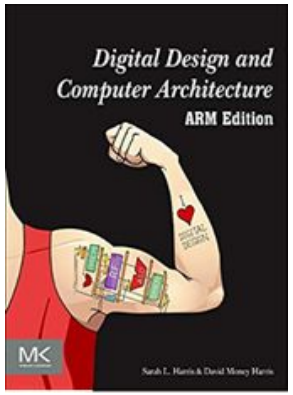
ECE 111 course overview

- Digital Circuit design using the hardware description (SystemVerilog) Language
- Use tools to simulate and implement circuits in the SystemVerilog
  - Using commercial CAD tools for ASIC/ Field Programmable Logic Arrays (FPGAs)
- Implementing designs and optimizing
  - Complex digital designs from algorithms to implementation
  - Hands-on projects
- The name “SystemVerilog” describes hardware at the same level as “Verilog”, but it adds a number of enhancements and improved syntax
- SystemVerilog files have a “.sv” extension so that the compiler knows that the file is in SystemVerilog rather than Verilog

ECE 111 course overview (cont'd)

- SystemVerilog Modeling styles
- Learn how to develop SystemVerilog testbench and perform RTL simulation
- SystemVerilog Synthesis Coding Guidelines through example
- Digital design timing and pipeling fundamentals
- Understand FPGA architecture and synthesis RTL code for FPGA
- Introduction to SystemVerilog Assertions
- EDA Tools :
  - Learn how to use Modelsim simulator to perform RTL simulation and debug waveforms
  - Debugging waveforms using Novas Verdi
  - Learn how to use Altera Quartus FPGA tool sets to perform synthesis, view RTL and post mapping nelist and review timing reports

Recommended (not mandatory) textbooks



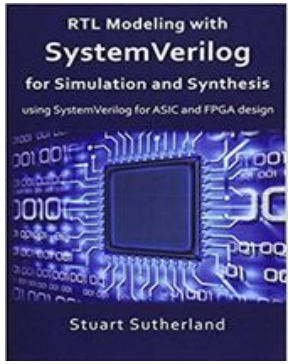
Digital Design and Computer Architecture: ARM Edition 1st Edition

Author : Sarah Harris and David Money Harris

ISBN-13: 978-0128000564

ISBN-10: 9780128000564

(We will mainly be using Chapter 3 (background) and Chapter 4 (System Verilog))



RTL Modeling with SystemVerilog for Simulation and Synthesis: Using SystemVerilog for ASIC and FPGA Design, 1st Edition

Author : Stuart Sutherland

ISBN-13: 978-1546776345

ISBN-10: 1546776346

IEEE SystemVerilog, LRM https://standards.ieee.org/standard/1800-2012.html
Accellera https://www.accellera.org/downloads/standards

This is a project-based class. Class time used to talk about SystemVerilog, specific examples and projects. The students are required to write HDL code for the projects and synthesize..

Lectures

- Lectures are based on slides – both great and bad
- Slides are great
  - Organizes the course material
  - Great addition to the textbook
  - Can use pictures and animation to explain
- Slides are bad
  - Well-known that instructors teach faster by slides
  - They make hard things look easier than they are
  - They can make you fall asleep
- I encourage you to be proactive and take notes
- I may use the board when/if necessary

Software

- **Intel Altera Quartus Prime**
  - Includes both RTL code synthesis, implementation and simulator tools
  - ModelSim-Altera FPGA edition simulator is part of the installation
  - We will give you the latest instructions on the version we will be using on Canvas
- **Quartus Prime Lite download and installation instructions will be published on Canvas and will be discussed by the TAs**
  - For Windows OS machines and MAC Machines we will have Canvas Instructions
- **We will have a SW setup date for you all on Zoom**

Class work

- Small weekly homework programming assignments (making sure you follow the class)
- 6 homework assignments
- Final project is a two part project. It will be evaluated on 2 criteria:
  - Best delay = number of clock cycles x clock period
  - Best area/delay = area x delay
- The two final projects, done in teams of 2 students

Course grading policy

- **Assignments: 40% of final grade**
  - There will be 6 mini projects / programming-based assignments. Which will include designing circuits by SystemVerilog, develop testbench, synthesis and simulation
  - Please turn in all assignments (attendance grade 5%)
  - Can pick your best 5 homework assignments (excluding one) to grade

- **Participation: 5% of final grade**
- **Project and presentation : 55% of final grade**
  - RTL code development, FPGA synthesis, Verification and Final Presentation
  - More information on the final project will be presented later in this quarter

- **More information on the final project will be presented later in this quarter**

Honor code

- The UCSD student conduct code
- https://students.ucsd.edu/sponsor/student-conduct/regulations/22.00.html
- Violations will be reported to the Student Conduct Office
- Accommodations available for students with disabilities, religious and ethnic holidays, and illness (with proper documentation)

Syllabus (Tentative)

Lec	Date	Subject	Assignment
1	09/26/2024	Introduction	
2	10/01/2024	Intro to System Verilog	Homework_1
3	10/03/2024	ASIC, FPGA, Logic Synthesis	
4	10/08/2024	System Verilog (1)	Homework_2
5	10/10/2024	System Verilog (2)	
6	10/15/2024	System Verilog Data Types and Statements	Homework_3
7	10/17/2024	Blocking and non-blocking assignments (1)	
8	10/22/2024	Procedural Blocks (1)	Homework_4
9	10/24/2024	Procedural Blocks (2)	
10	10/29/2024	RTL Programming Statements (1)	Homework_5

Syllabus (Tentative)

11	10/31/2024	RTL Programming Statements (2)	
12	11/05/2024	Finite State Machine (1)	Homework_6
13	11/07/2024	Finite State Machine (2)	
14	11/12/2024	SHA 256 algorithm	Final Project part 1
15	11/14/2024	Memory Modeling, Packed and Unpacked Arrays	
16	11/19/2024	Delay optimization	Final Project part 2
17	11/21/2024	Area optimization	
18	11/26/2024	Project discussions with Prof and TAs	
	11/28/2024	Thanksgiving day	
19	12/03/2024	Project discussions with Prof and TAs	
20	12/05/2024	Final project FAQ with Prof and TAs	
	12/13/2024	Final project report Due	
Note that the homeworks are due a week after their release			

About you

1. Your department
2. Undergrad/graduate student? Year?
3. Have you taken a HDL course earlier? If so, please describe a little more.
4. What is your aim for taking the course?
5. What do you expect to learn in this course?
6. What qualities you value most in an instructor?
7. How can I help you best in learning the course material?

Why this class?

Why digital design is important?

- Modern lives depend on digital technology

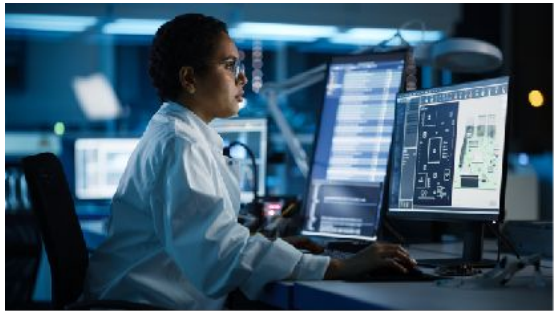
- Digital technology depends on digital design

Why digital design is important?

There's an **insatiable** demand for faster and more efficient computers!

Gamers

Software Developers



Scientists

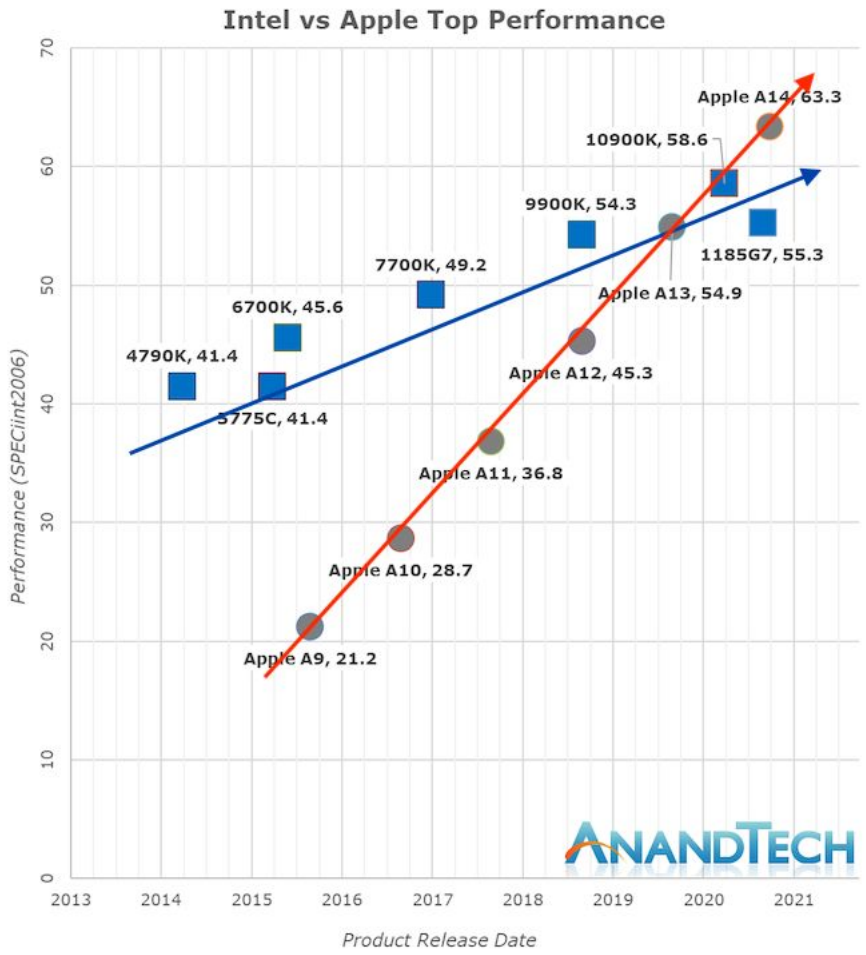
Students

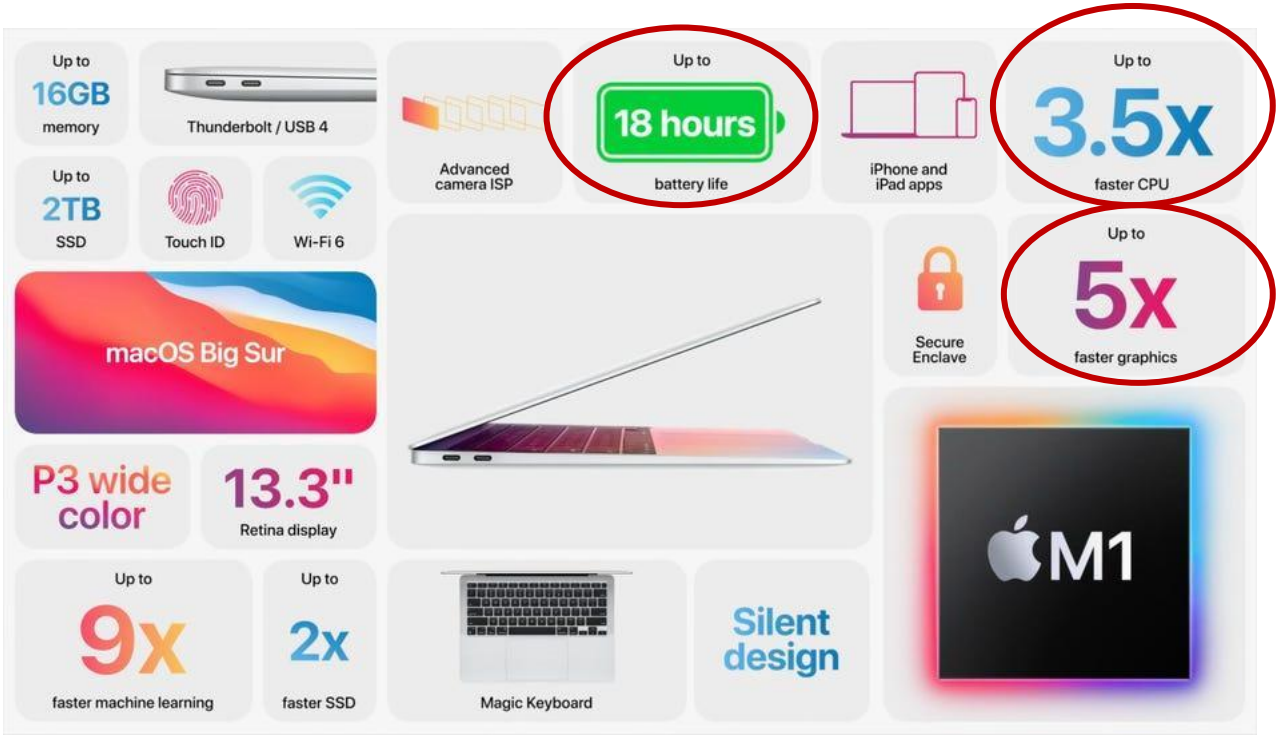
Athletes

Artists

(Adobe stock)

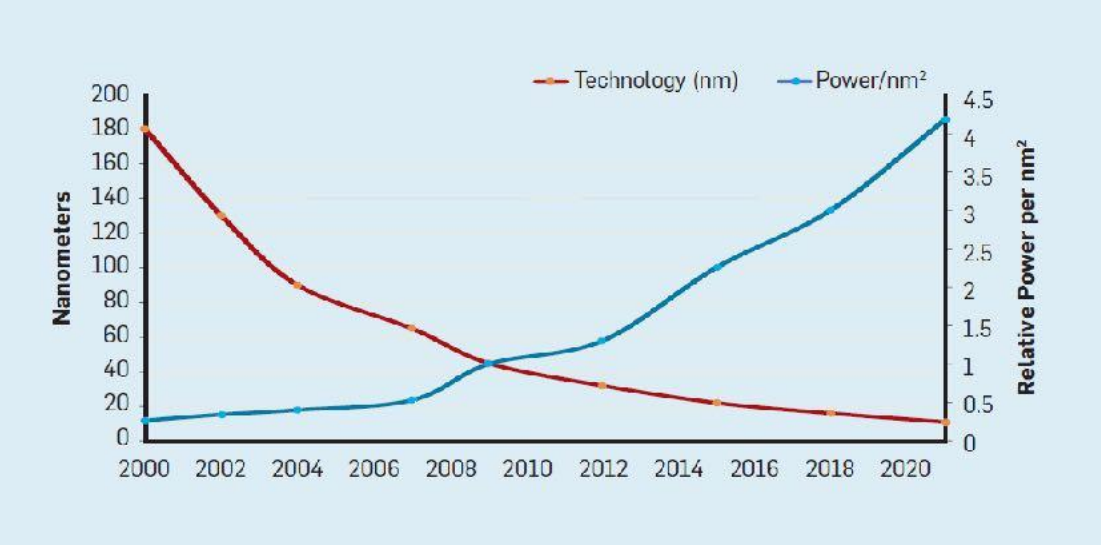
Performance race





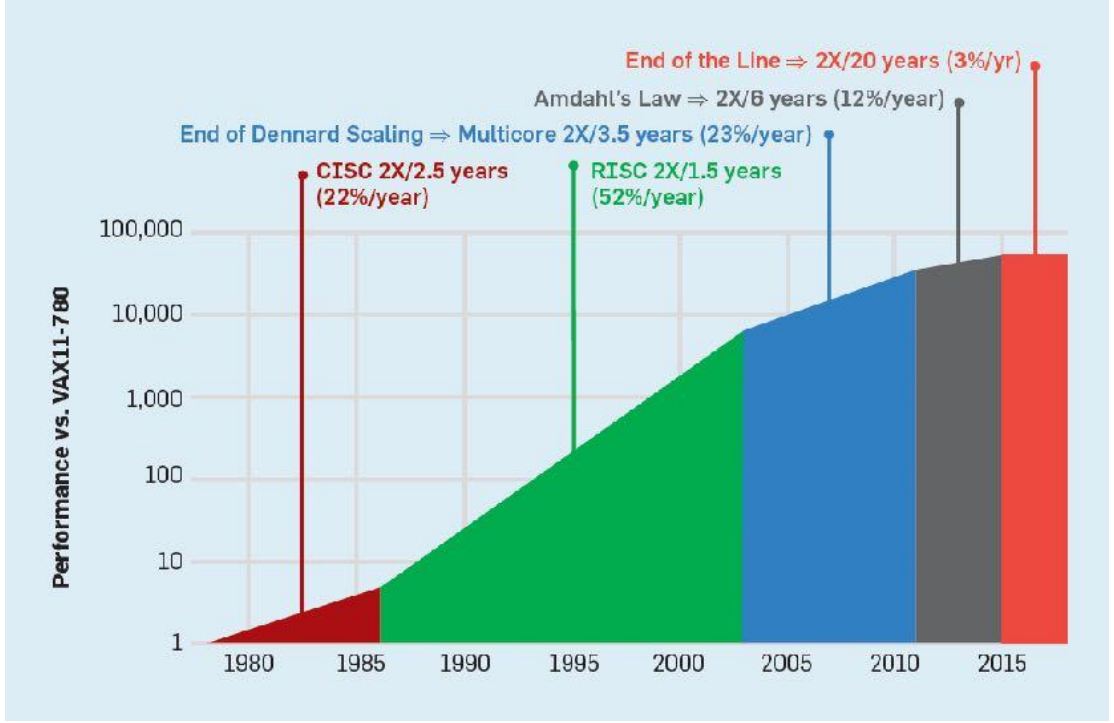
A new golden age in computer architecture

Transistors size and power per mm^2



- Performance scaling must continue!
- There's not enough performance gain by simply making the transistors smaller
- We need to design digital circuits for higher performance and efficiency (this course!)

Computer performance scaling



Hennessy and Patterson, CACM 2019)

Big tech shift to vertical integration

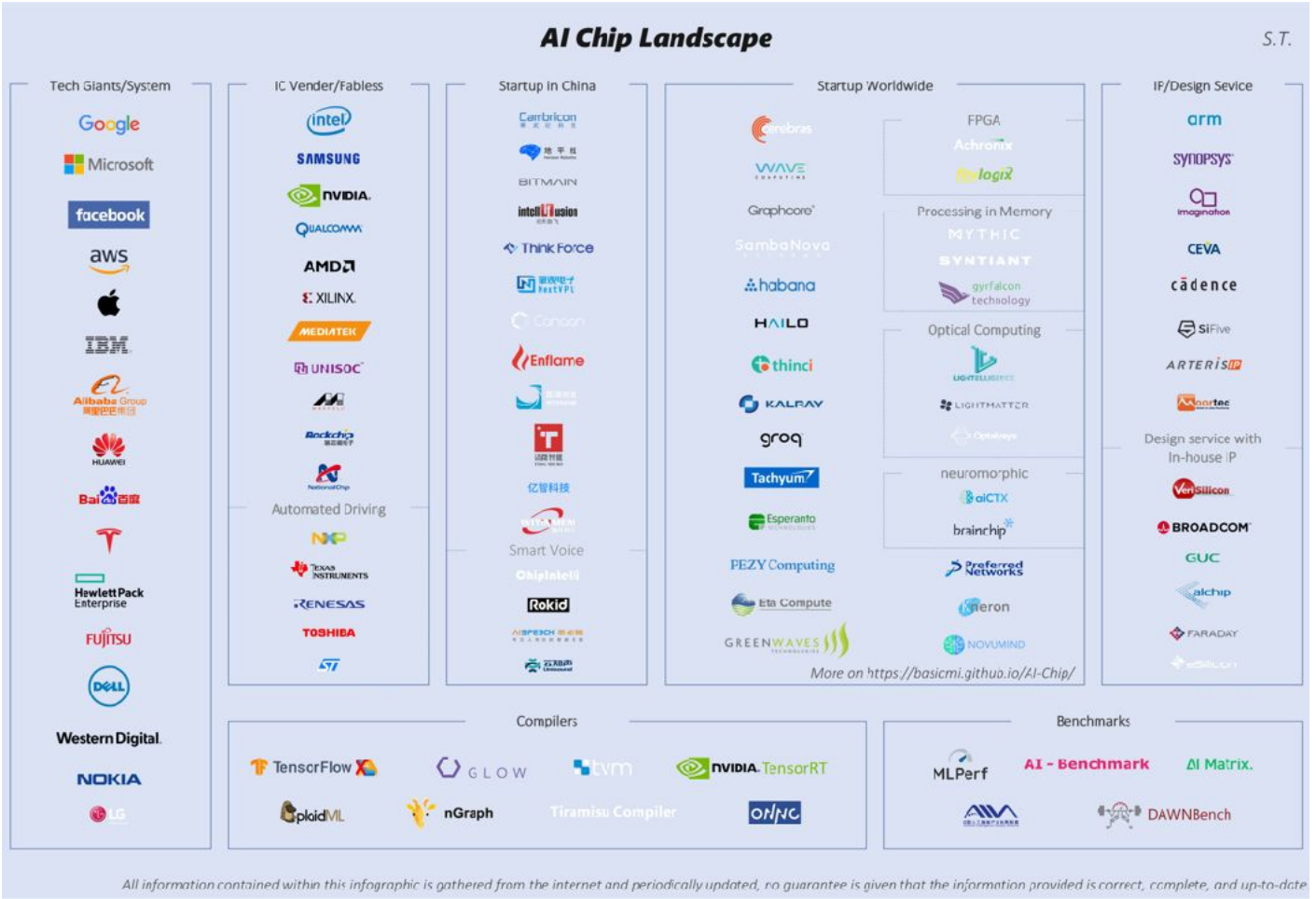
Company	Platforms	Suppliers	In-House Opportunity
Amazon	Amazon.com, AWS, Alexa, Kindle	Intel, Broadcom, Nvidia, ARM	Graviton2 – in-house ARM chips. Annapurna – In-house DPU, in-house AI Chip
Apple	iPhone, Mac, iPad Wearables, Services	Intel, Broadcom, Qualcomm, Skyworks, ARM	M1 Chip + Ax Chip Platform. In-house Modem (INTC's)
Facebook	Facebook, Instagram, WhatsApp, Oculus	Intel	ARM chips?
Google	Search, YouTube, GCP, and Other	Intel, Nvidia, Qualcomm	TPU chip, Custom ARM chip for phone
Microsoft	O365, Windows, Azure, Xbox, Surface, LinkedIn	Intel, Xilinx, Nvidia	ARM chip for Surface, Catapult

https://www.fabricatedknowledge.com/p/the-tech-monopolies-go-vertical?s=r

AI and surge in chip start-ups

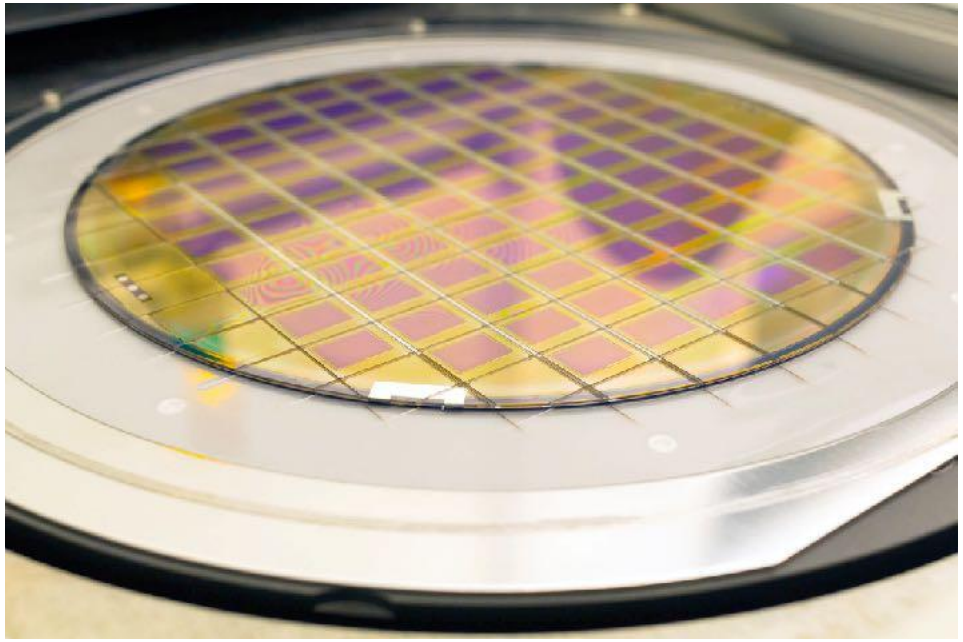
Cost of training the AI model

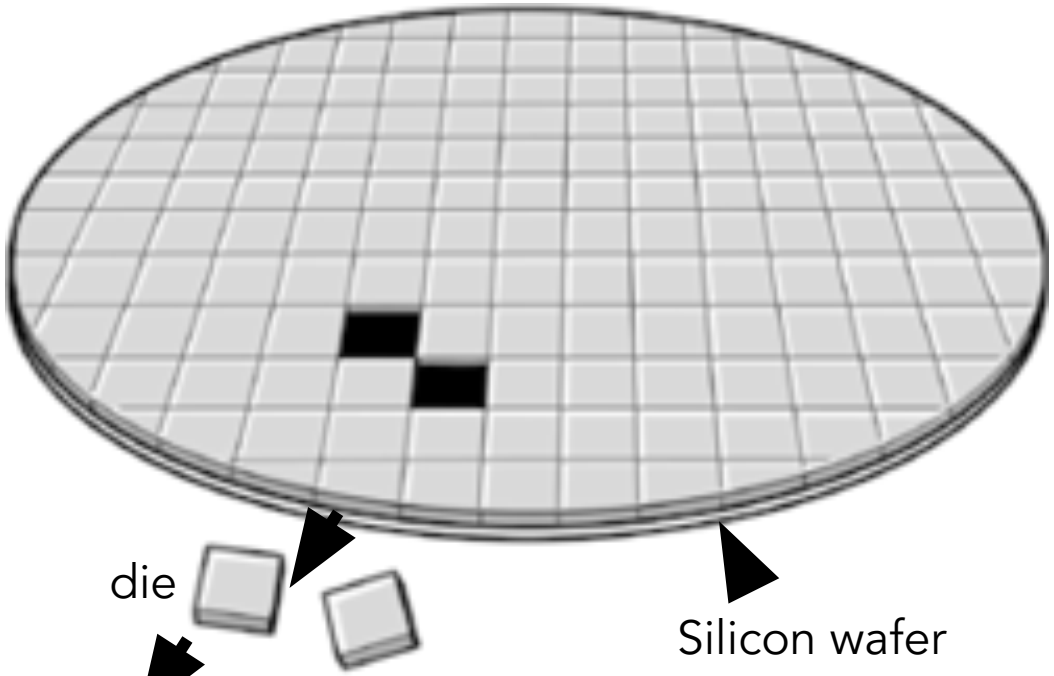
Model	Year	Energy consumption (kWh)	Carbon footprint (lbs of CO2)	Cloud compute cost (USD)
Transformer (213M parameters)	2017	201	192	$289-$981
BERT (110M parameters)	2018	1,507	1,438	$3,751-$12,571
Transformer (213M parameters) w/ neural architecture search	2019	656,347	626,155	$942,973-$3,201,722

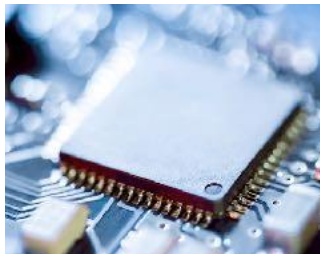


https://semiwiki.com/ | MIT Technology Review

Silicon wafers and chips







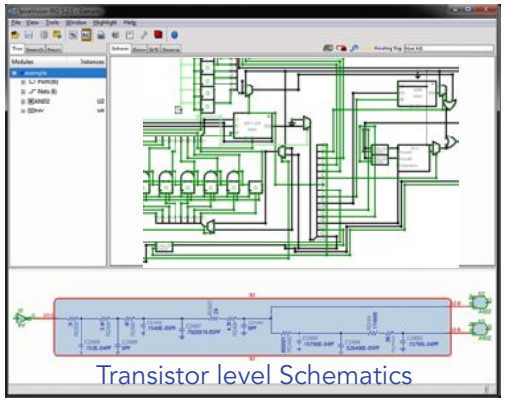
VLSI chip or integrated circuit (IC)

Chip design process

Step-1 : Draw Logic Circuit Schematics for Multiplier

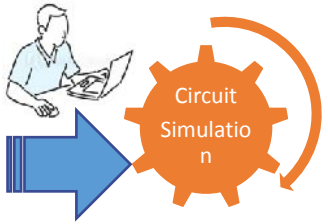
Hardware Design Requirement & Specification (Design Multiplier)

Gate level Schematics (Concept Design)



Transistor level Schematics

Step-2 : Pre-Layout Circuit Simulation



Circuit Simulation Results (Waveform)

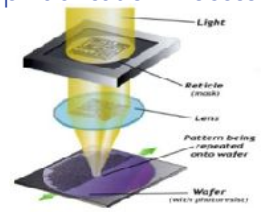
Update Schematics

No

Does this work as a Multiplier ?

Multiplier Chip

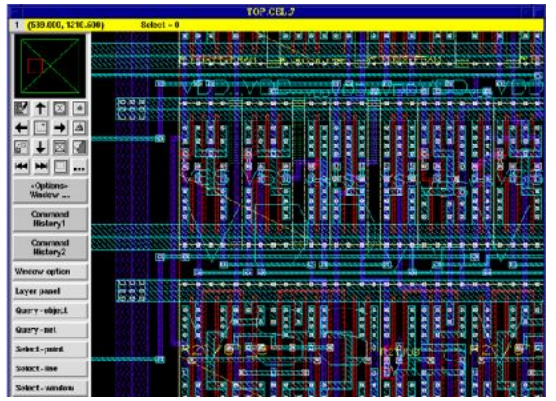
Chip Fabrication Process



Step-5 : Send Layout to Foundry for Chip Fabrication

Yes

Final Transistor Level Custom Layout (Actual Design)



Yes

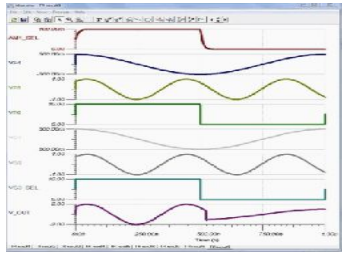
Step-3: Draw Transistor Level Layout

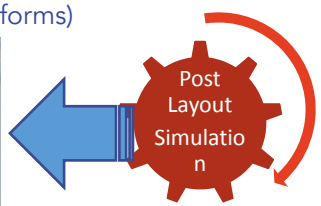
Does this work as a Multiplier and meet requirements ?

No

Update Layout

Characterization Results (Waveforms)





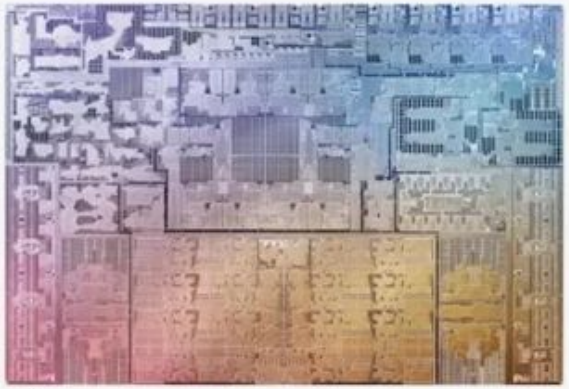
Step-4: Post-Layout Simulation

Modern chips have an unfathomable complexity

~16 billion transistors!

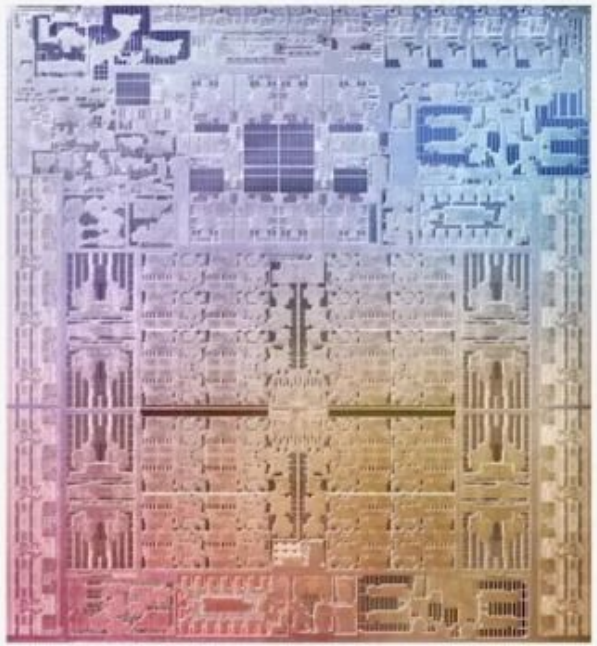
M1

~34 billion transistors!



M1 Pro

~57 billion transistors!



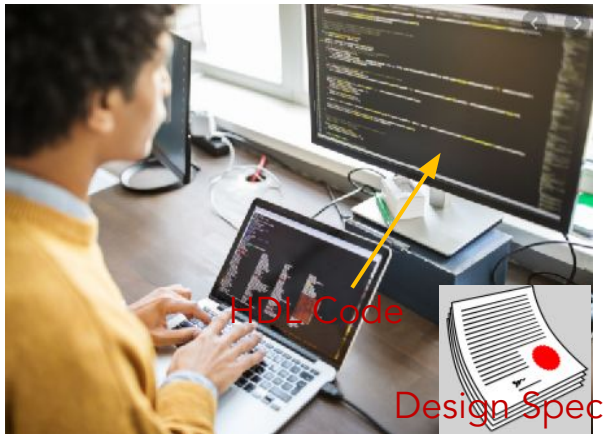
M1 Max

https://www.phonearena.com/

Hardware Description Language (HDL)?

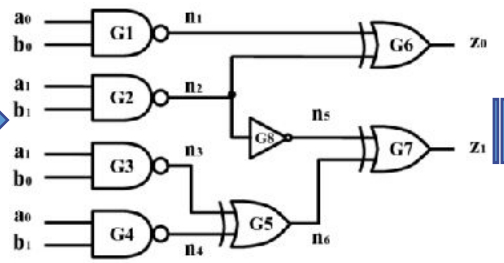
Hardware Description Language (HDL) to reduce chip complexity

SystemVerilog HDL Code to Model Hardware Design



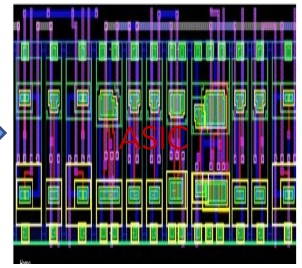
Logic Synthesis

Gate Level Circuit

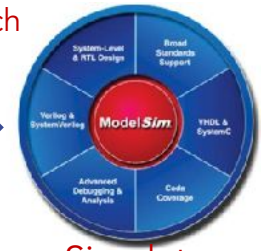


Physical Synthesis

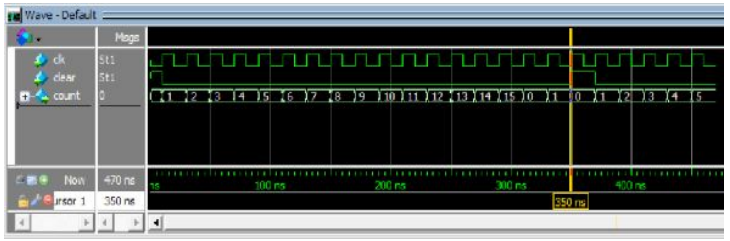
Transistor Level Layout



System Verilog Design & Testbench Code



Simulator

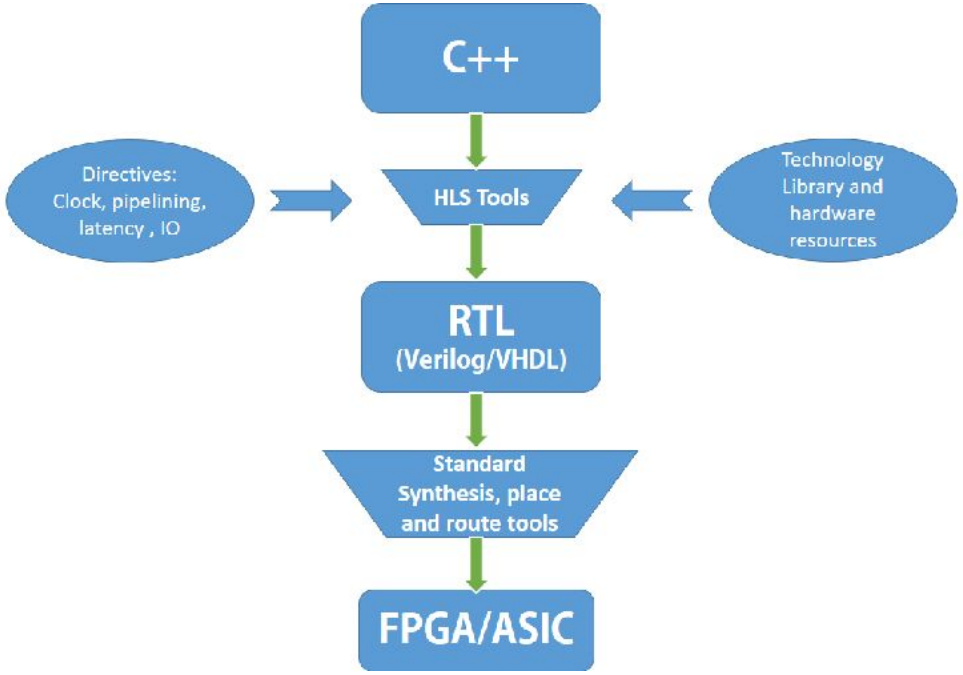


Waveform to Virtually Confirm Hardware Design behavior before manufacturing

- HDLs allow concisely expressing the logic circuit behavior without dealing with the complexity of gate or transistor level implementation
- Synthesis tools can automatically convert HDL code to gate-level circuit and then to transistor-level layout
- Logic simulators can perform simulation of HDL code to test circuit behavior before the hardware is manufactured

Raising the abstraction level is hard!

- High-level synthesis (HLS) aims to automate digital design – convert C/C++/SystemC specification to its VHDL/Verilog implementation
- Experience so far with HLS tools suggests they are not as efficient as manually designed digital circuits
- Digital design is hard but automating it seems even harder!

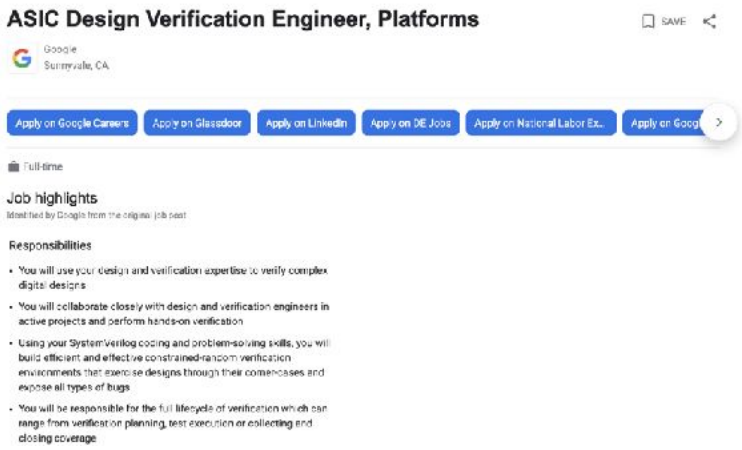


https://thedatabuo/hls-ins.itroduction

What makes digital designs so hard?

- Notion of a ‘clock’
  - Digital logic and technology influences clock frequency, which in turn influences latency and throughput
  - No equivalent concept in software
- Software is typically sequential, but all hardware components are operating in parallel
- Resources optimization
  - Area
  - Speed
  - Power

Digital designers are in high demand



ASIC Design Verification Engineer, Platforms

Google
Sunnyvale, CA

Apply on Google Careers | Apply on Glassdoor | Apply on LinkedIn | Apply on DE Jobs | Apply on National Labor Ex.. | Apply on Goog

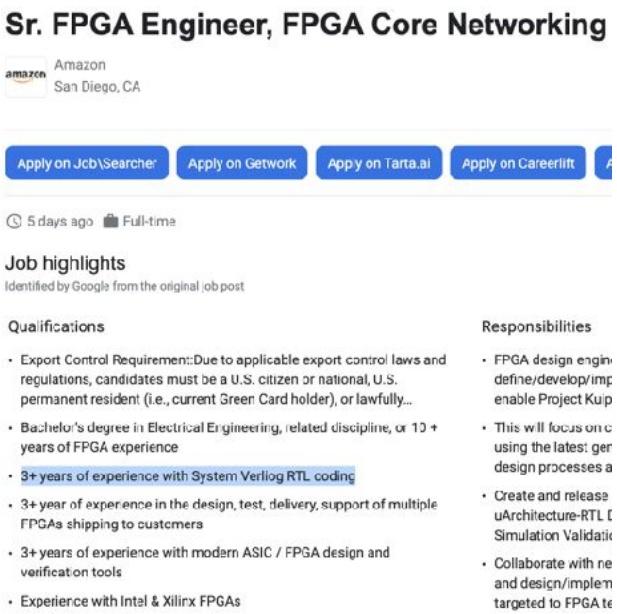
Full-time

Job highlights

Identified by Google from the original job post

Responsibilities

- You will use your design and verification expertise to verify complex digital designs
- You will collaborate closely with design and verification engineers in active projects and perform hands-on verification
- Using your SystemVerilog coding and problem-solving skills, you will build efficient and effective constrained-random verification environments that exercise designs through their cornercases and expose all types of bugs
- You will be responsible for the full lifecycle of verification which can range from verification planning, test execution or collecting and closing coverage



Sr. FPGA Engineer, FPGA Core Networking

Amazon
San Diego, CA

Apply on Jcb\Searcher | Apply on Getwork | Apply on Tarta.ai | Apply on Careerlift

5 days ago | Full-time

Job highlights

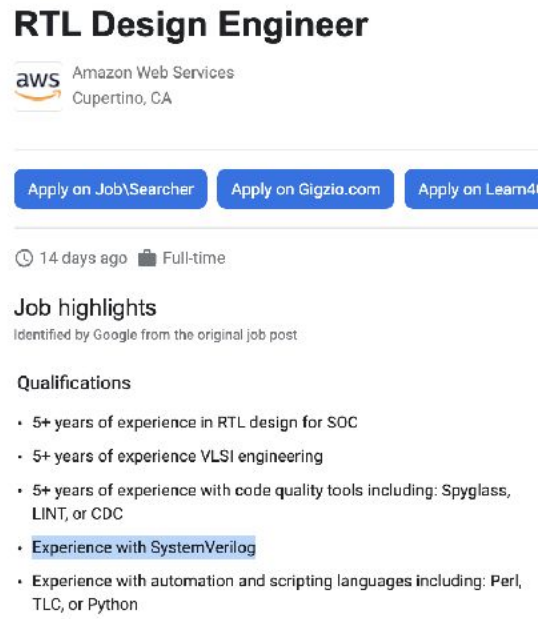
Identified by Google from the original job post

Qualifications

- Export Control Requirement::Due to applicable export control laws and regulations, candidates must be a U.S. citizen or national, U.S. permanent resident (i.e., current Green Card holder), or lawfully...
- Bachelor's degree in Electrical Engineering, related discipline, or 10 + years of FPGA experience
- 3+ years of experience with System Verilog RTL coding
- 3+ year of experience in the design, test, delivery, support of multiple FPGAs shipping to customers
- 3+ years of experience with modern ASIC / FPGA design and verification tools
- Experience with Intel & Xilinx FPGAs

Responsibilities

- FPGA design engin
define/develop/imp
enable Project Kuip
- This will focus on c
using the latest: ger
design processes a
- Create and release
uArchitecture-RTL
Simulation Validati
- Collaborate with ne
and design/implem
targeted to FPGA te



RTL Design Engineer

Amazon Web Services
Cupertino, CA

Apply on Job\Searcher | Apply on Gigzio.com | Apply on Learn4

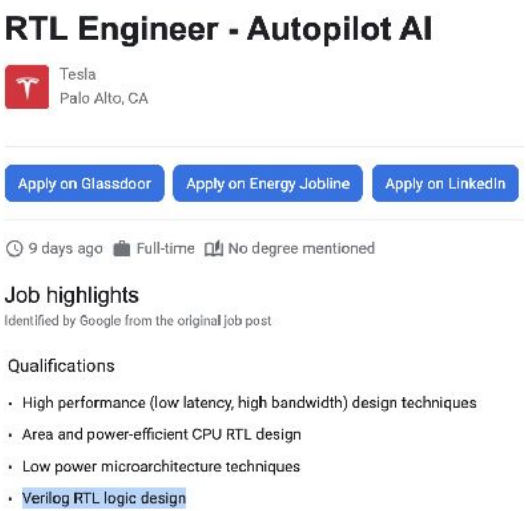
14 days ago | Full-time

Job highlights

Identified by Google from the original job post

Qualifications

- 5+ years of experience in RTL design for SOC
- 5+ years of experience VLSI engineering
- 5+ years of experience with code quality tools including: Spyglass, LINT, or CDC
- Experience with SystemVerilog
- Experience with automation and scripting languages including: Perl, TLC, or Python



RTL Engineer - Autopilot AI

Tesla
Palo Alto, CA

Apply on Glassdoor | Apply on Energy Jobline | Apply on LinkedIn

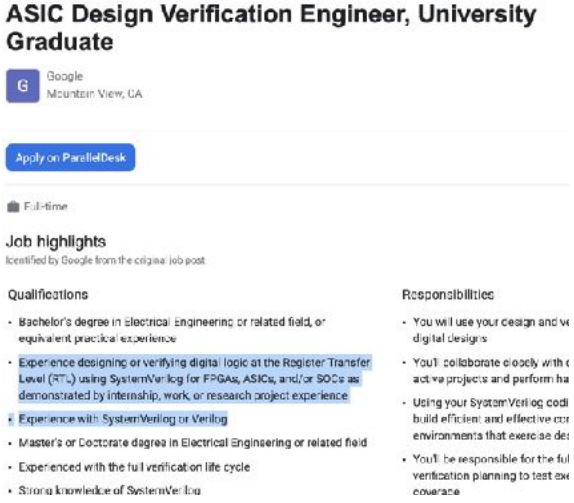
9 days ago | Full-time | No degree mentioned

Job highlights

Identified by Google from the original job post

Qualifications

- High performance (low latency, high bandwidth) design techniques
- Area and power-efficient CPU RTL design
- Low power microarchitecture techniques
- Verilog RTL logic design



ASIC Design Verification Engineer, University Graduate

Google
Mountain View, CA

Apply on ParallelDesk

Full-time

Job highlights

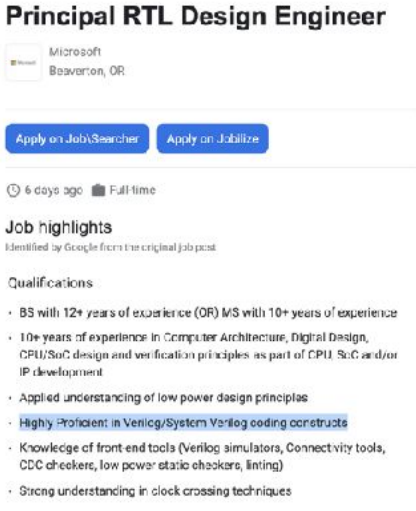
Identified by Google from the original job post

Qualifications

- Bachelor's degree in Electrical Engineering or related field, or equivalent practical experience
- Experience designing or verifying digital logic at the Register Transfer Level (RTL) using SystemVerilog for FPGAs, ASICs, and/or SOCs as demonstrated by internship, work, or research project experience
- Experience with SystemVerilog or Verilog
- Master's or Doctorate degree in Electrical Engineering or related field
- Experienced with the full verification life cycle
- Strong knowledge of SystemVerilog

Responsibilities

- You will use your design and ve
digital designs
- You'll collaborate closely with
active projects and perform ha
- Using your SystemVerilog codi
build efficient and effective co
environments that exercise de
- You'll be responsible for the ful
verification planning to test ex
coverage



Principal RTL Design Engineer

Microsoft
Beaverton, OR

Apply on Job\Searcher | Apply on Jobilize

6 days ago | Full-time

Job highlights

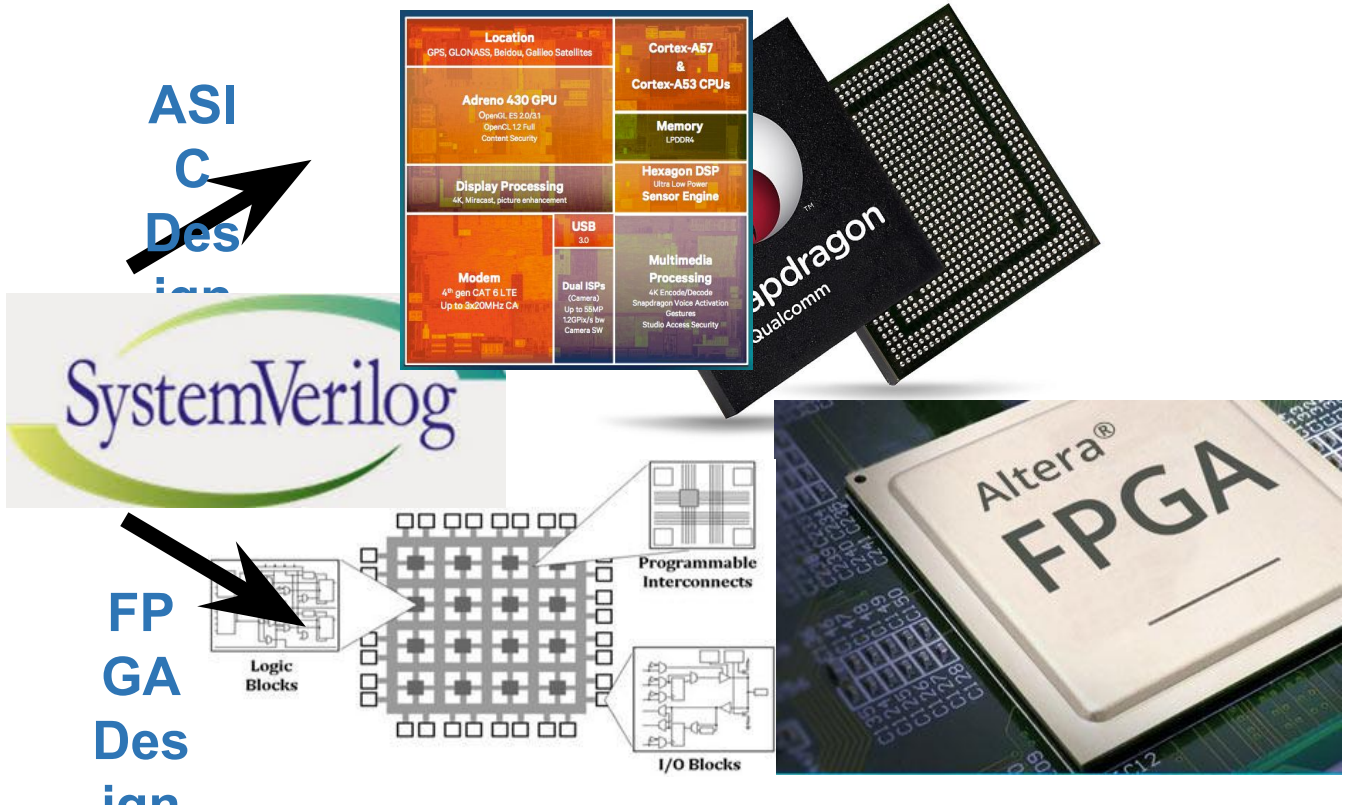
Identified by Google from the original job post

Qualifications

- BS with 12+ years of experience (OR) MS with 10+ years of experience
- 10+ years of experience in Computer Architecture, Digital Design, CPU/SoC design and verification principles as part of CPU, SoC and/or IP development
- Applied understanding of low power design principles
- Highly Proficient in Verilog/System Verilog coding constructs
- Knowledge of front-end tools (Verilog simulators, Connectivity tools, CDC checkers, low power static checkers, linting)
- Strong understanding in clock crossing techniques

Why Verilog/SystemVerilog?

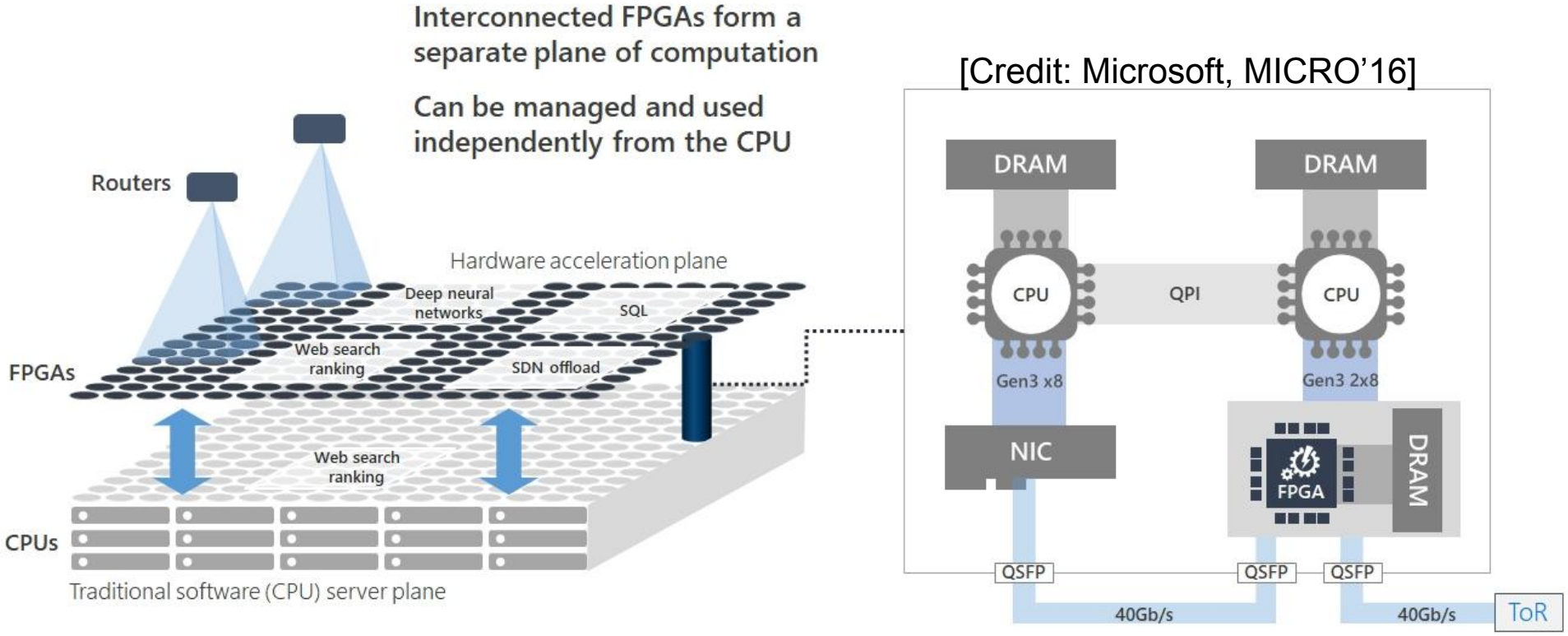
- SystemVerilog is used for both ASIC design and FGPA prototyping
- Most EE jobs are Verilog/SystemVerilog based chip designs



Why Verilog/SystemVerilog?

- Emergence of FPGA Cloud

  Example: Microsoft's Catapult Project deployed worldwide

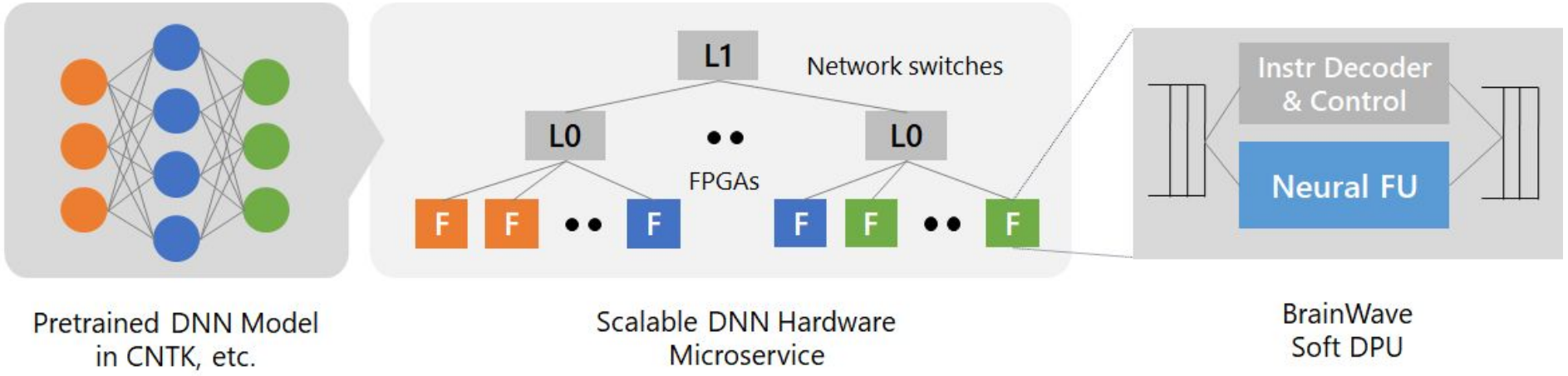


All Soft IP's such as CPU, DRAM Controller, Interconnects, NIC are all designed using Verilog/SystemVerilog !!!

Why Verilog/SystemVerilog?

- Emergence of FPGA Cloud

Example: Microsoft's Project BrainWave



Each FPGA implements many Soft DPUs using Verilog/SystemVerilog !!!

[Credit: Microsoft, Hot Chips'17]

Why Verilog/SystemVerilog?

- With more and more FPGA based Clouds emerging, Verilog/SystemVerilog usage is further increasing beyond ASIC chip designs

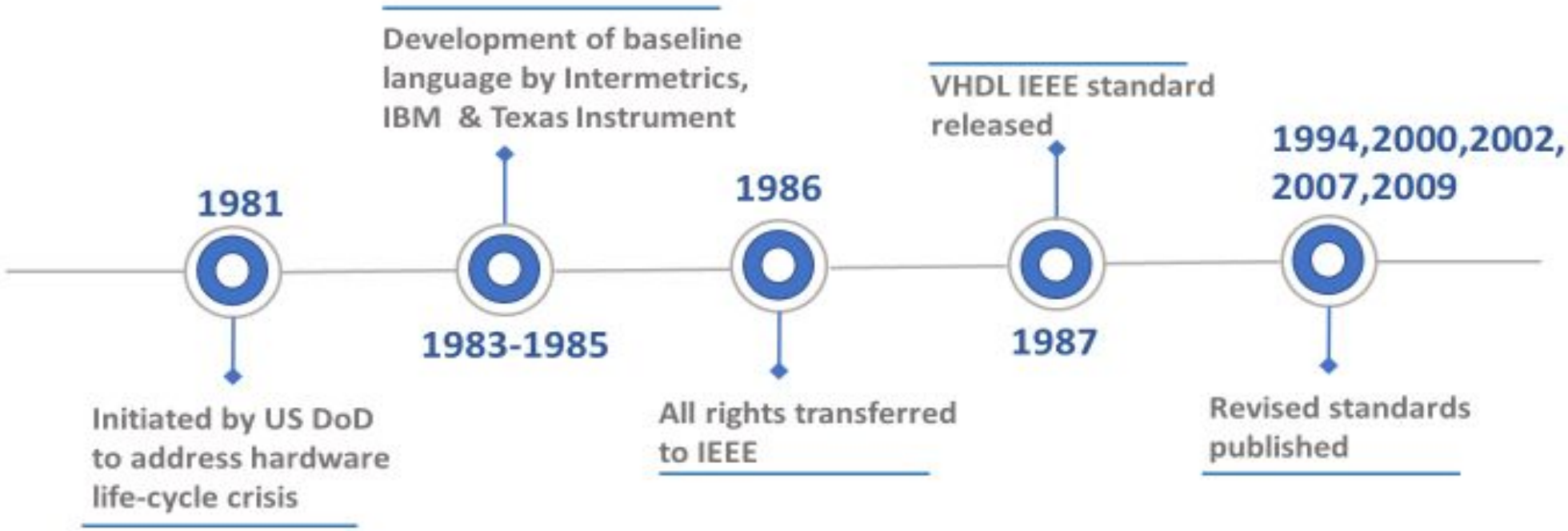
Hardware Description Language (HDL)

- **Instead of describing hardware in form of gates, transistors and generating layouts, HDL allows designers to describe the behavior of design using a 'C'-like programming language**
  - Digital design engineer will identify which portions are combinational logic, which are sequential logic or finite state machines and so forth, then write HDL code for each portion
  - HDL will allow to verify and visualize systems and designs much efficiently than drawing and analyzing complete schematics

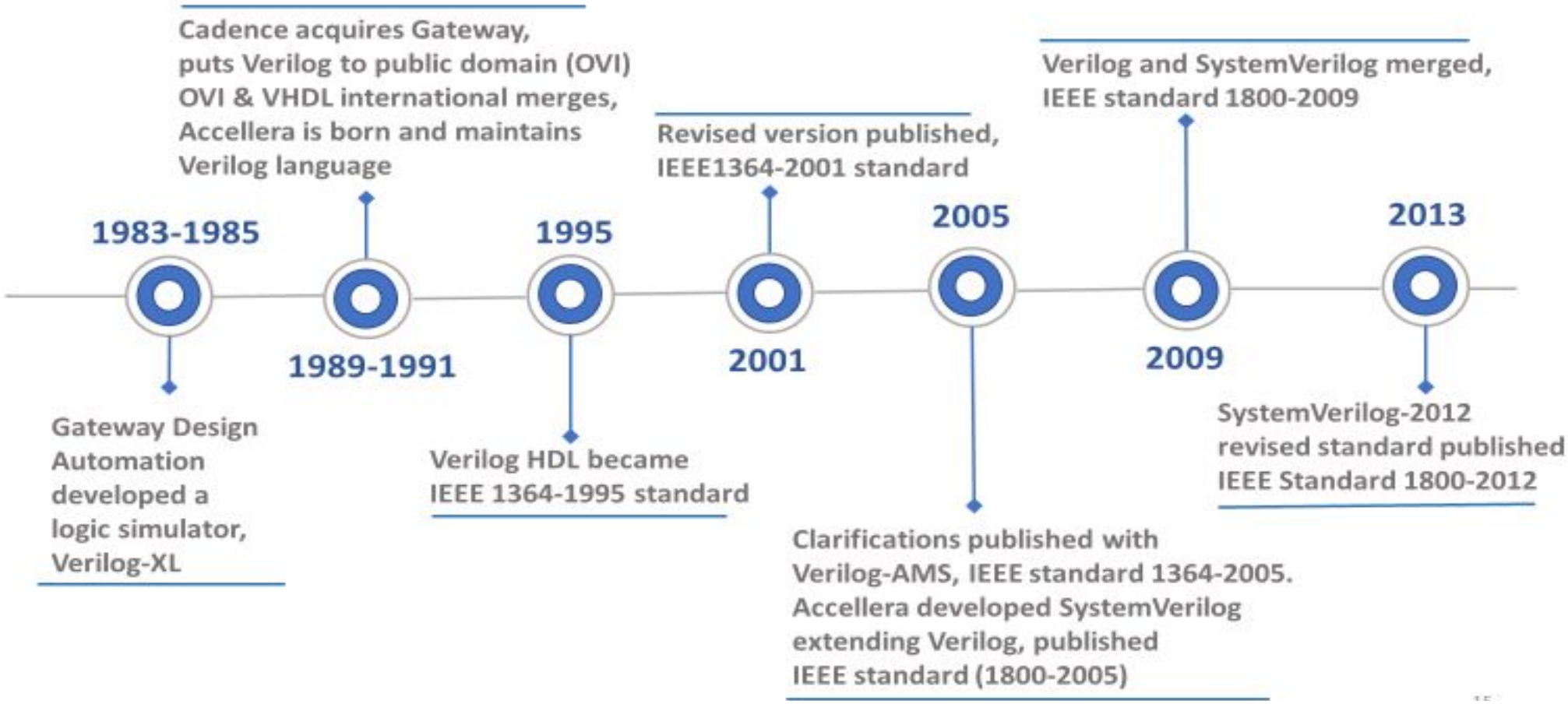
- **HDL is used for both simulation and synthesis**
  - Logic Synthesis to convert HDL code to actual logic gates
  - Simulation to test system on computer before actual hardware is built and manufactured

- **There are two main Hardware Design Languages**
  - VHDL (**V**ery **H**igh **S**peed **I**ntegrated **C**ircuit **H**ardware **D**escription **L**anguage)
  - SystemVerilog

VHDL evolution

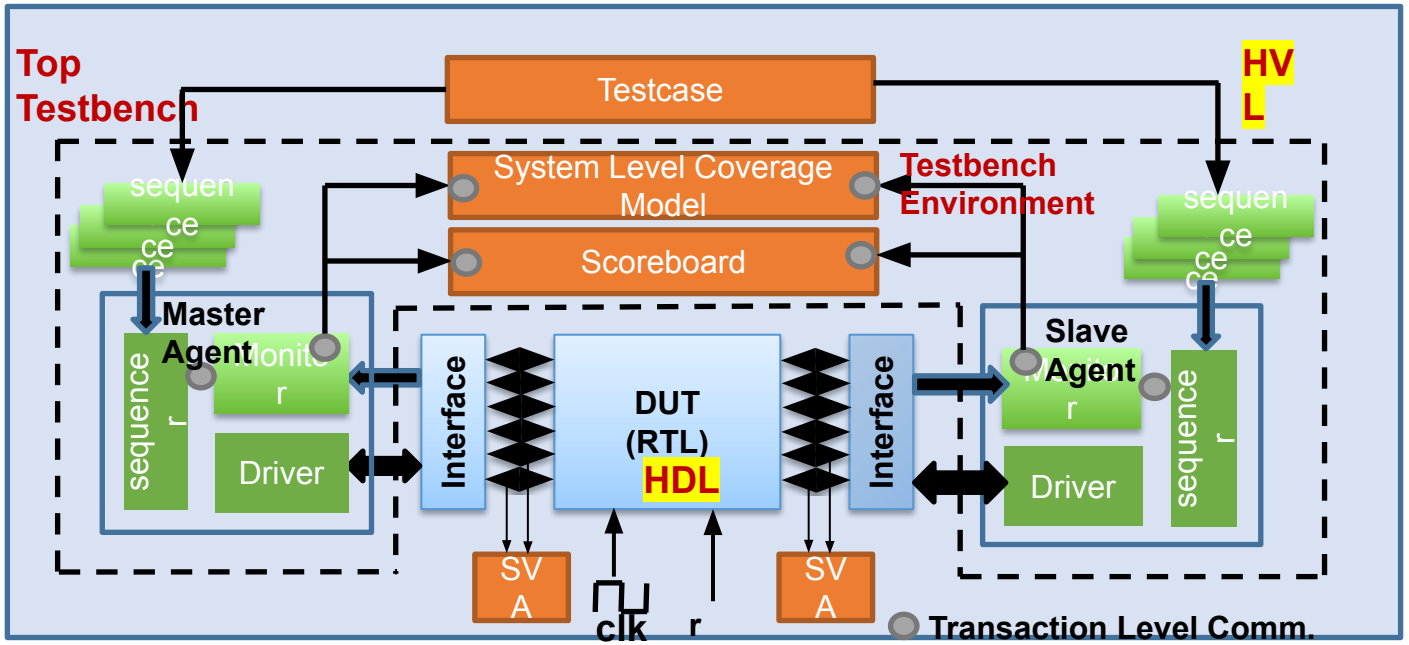


Verilog/SystemVerilog Evolution



Why SystemVerilog over other languages?

- One language for design and verification !!!
  - SystemVerilog is both H**D**L and H**V**L. Seamless data flow from testbench world to design world !
  - Design described using synthesizable constructs. More concise representation of design **compared to VHDL**
  - Re-usable, scalable, constraint and coverage driven verification environment using C++ like constructs (**not in VHDL**)
  - Advance methodologies such as UVM, VMM for verification and enables hardware acceleration
- Can interface with C and System-C languages.
- Provides extensions to create Analog Mixed Signal Models (Verilog AMS)



- HDL is used for developing design code (RTL)

- HDL code is synthesizable and is part of final product
- Non-OOP based constructs

- HVL is used for developing testbench to stimulate and verify design

- HVL code is non-synthesizable and is not part of final product
- OOP based constructs

What can be done using SystemVerilog?