Dinesh Reddy Chinta

+1 (904) 678 2615 | dineshc@utexas.edu | https://chintadinesh.github.io

EDUCATION

UT Austin, *MS in Computer Systems (ACSES track)* | Austin, USA GPA: 3.67 / 4.0 May 2023
IIT Hyderabad, *B.Tech in EE, CSE double major* | Hyderabad, India GPA: 8.78 / 10 May 2019

Courses: Advanced OS | Parallel Computer Architecture | Cross-Layer ML | Edge A.I | RTOS Lab | Advanced MCU | SOC-Design | Computer Architecture | Data Structures & Algorithms | Applied Machine Learning
Academic Jobs: Introduction to Embedded Systems (EE319K), Digital Logic Design (EE316)

EXPERIENCE

Cadence Design Systems, *Software Engineer*, Hardware System Verification | (San Jose, USA) May 2023 - now

- Developed graph algorithms for **ICG-based clock transformations** in Precompiler to enhance emulation performance.
- Implemented template meta and generic programming techniques in internal **C++ libraries**.
- Currently developing **Bit-stream** generation modules for Athena architecture, Palladium's successor.

Apple, *Symtem Architecture Modelling Intern*, Vision Pro | (Sunnyvale, USA) May 2022 - Aug 2022

- Automated a **SystemC and Python**-based use-case modeling framework by abstracting HW and tasks into XML files, enabling seamless system configuration instead of hardcoded architecture.
- Implemented a fine-grained **dynamic task scheduler** to replace a coarse-grained average-based scheduler, improving latency and power modeling in the SystemC framework.

Xilinx, *Silicon CAD Engineer* | (Hyderabad, India) June 2019 - Aug 2021

- Implemented and maintained Python, Perl, and TCL scripts for creating and managing **Perforce workspaces**, used daily by PnR design engineers. I also developed regression scripts in Python.
- Led the development of the **Seras, an internal GUI platform**, from its early stages, validating new features, debugging, and resolving issues to support the 'Synthesis Placement & Routing' design flow. This infrastructure enabled Cadence's Innovus tool adaptation for Xilinx's Place and Route (PnR) tasks.
- **Project updates to a dashboard**: Wrote pre- and post-synthesis TCL scripts to extract key design metrics and aggregated them in a MongoDB, enabling dashboard generation for design teams and upper management.

SKILLS

Languages	C++, C, Python, Tcl, Perl, Bash
Software	Linux, Git, Perforce, Vim

PROJECTS

Advanced Operating Systems - *UT Austin* Jan 2023 - May 2023

- **Program Measurement and mmap:** Explored building, booting, and running the Linux kernel on the QEMU-KVM hypervisor. Showcased debugging techniques for investigating kernel data structures through practical examples, and examined key aspects of VMs, including timer management, device management, and page table management.
- **User-level File System based on FUSE:** Implemented a cached in-RAM file system to accelerate networked file systems. Single-byte random access workloads showed a 10x performance improvement over NFS.
- **Program loading:** Implemented ELF loaders to map program segments and set up the stack to transfer control to the child process. Developed a demand-mapped loader variant that minimizes memory usage by trapping signals.
- **Project - IO_URING:** Demonstrated the efficacy of IO__URING, an asynchronous I/O framework, in file-copy utilities by reducing the time taken to complete the task by 88% when recursively copying files in a directory, 80% for 32MB files, and 72% for 1GB files.

Parallel Computer Architecture - *UT Austin* Aug 2022 - Dec 2022

- **Directry-based Cache Coherency:** Simulated MESI cache coherence protocol in c++.
- **Memory Reliability through Replication:** Explored hardware-software techniques for replication in caches and DRAM. For caches, we proposed a cache-coherency protocol with a new SlaveModified (SM) state to improve capacity and reduce performance costs. For memory replication, we introduced a page-mapping technique with a coherency protocol and map-chaining strategy to support multiple replications. We analyzed performance gains, latency, memory, power, and network overhead, highlighting trade-offs based on use cases.

SOC Design - *UT Austin* Aug 2021 - Dec 2021

- **Tiny-Yolo on CPU:** Improved inference time from 11s to 6s with OpenMp and Fixed-Point computations (code).
- **HW-SW co-simulation in SystemC:** Simulated GEMM interacts with QEMU-SystemC environment to run Tiny-yolo.
- **Vivado HLS:** Synthesized a memory-partitioned run-time-optimized GEMM accelerator.
- **GEMM IP on Zynq-ultra96:** Tiny-yolo on ZynqUltra96 achieved 16s inference time.

Achievements

Emerging Star of the FPGA Product Development Xilinx, 2020
Above & Beyond team award Xilinx, 2021