Howard Fu

817 W 30TH ST, LOS ANGELES, CA 90007 |+1 (302) 563-6256 | huayufu@usc.edu | http://huayufu.xyz

SUMMARY

Master's Student in Computer Engineering will graduate in May 2018, specializing in ***Digital VLSI Design/Verification*** and ***Computer Architecture*** with added emphasis on ***Object Oriented Programming*** and ***Embedded Systems*** and insight on *Engineering Leadership*. Proven ability to innovate and deliver results. Strong desire to be continuously learning. Thrive in challenging, dynamic environments, applying the full spectrum of proven capabilities, to make key contributions.

EDUCATION

University of Southern California, Los Angeles, CA May 2018
Master of Science, Computer Engineering, Digital VLSI and Computer Architecture GPA: 3.61
University of Delaware, Newark, DE May 2016
Bachelor of Electrical Engineering, Minor in Computer Science GPA: 3.5

SKILLS

Digital design: schematic, layout, Power Optimization, logic effort, STA, Asynchronous, ASIC, Full Customize, FPGA
Design tools: Xilinx ISE, Cadence Virtuoso, Design Compiler, IC Compiler II, ModelSim, HSpice
Programming: VHDL, Verilog, C/C++, Python, Perl, Tcl, Assembly

EXPERIENCE

Technical Intern, ***R&D, ECO team, Synopsys*** [C++, Tcl, IC Compiler II, gdb, Perforce, EDA] May - August 2017

- Proposed base layer density aware model for eco/spare cell placement to reduce the turnaround time by avoiding base layer density violations in earlier stage at backend design flow
- Implemented the model in *IC Compiler II* on remote *Unix server* with *version control tools* and *gdb,* tested the model with the ECO apps and Tcl test scripts, successfully demonstrated the correctness of the model
- Created high efficiency and high accuracy place-able region locating service with the visualization on *ICC II GUI*

Project Leader, *General-Purpose CPU Full-customize Design Project, USC* [Python, Cadence Virtuoso] March- April 2017

- Designed and verified a fully functional 16-bit general purpose pipelined CPU, featuring 512-bit SRAM and 16-byte register file, supports add/sub, mul, and/or, SFL/SFR, and store/load instructions.
- Programmed frontend and backend python code for instruction decoding and golden result generation
- Conducted in *schematic design*, *layout drawing*, performed *Power & Delay optimization* and employed *power gating* and *dynamic logic* techniques

Design Specialist, *Digital Neuron VLSI Design, USC* [Cadence Virtuoso] October - December 2016

- Designed a special-purpose digital circuit that mimics a neuron performed *logic* and *timing verification*
- Won *third place* in Design Contest with the area-delay product of 695 (class average 7000)

Design Specialist, *Music Visualizer Embedded System (SoC) Design, UD* [Xilinx, FPGA, VHDL, C, Python] February - May 2016

- Designed *FPGA*-based Music Visualizer with VGA output using *Xilinx IDE, EDK, and SDK*
- Created *VHDL* peripherals for fetching audio input, programmed audio processing and visualization algorithm running in ARM-based IP-core in *FPGA* board, and programmed controller to control Visualizer through *UART* port
- Empowered the *FPGA* board to output smooth patterns to VGA port, with two different display mode

PROJECTS

Asynchronous Design of Network-On-Chip (NoC) with FEC [SystemVerilog, modelsim]:

- Designed a tree-structured NoC with router, arbiter, merge and hamming code module in RTL level
- Generated gate level design through synthesis tool with *PCHB* templates

Dual-clock Asynchronous FIFO [Verilog, ModelSim]:

- Designed Dual-clock FIFO for cross clock domain communication with pointer synchronizer, featuring gray code converter to avoid data incoherence

Operating System Course Project [C, Multi-Threading, Linux, gdb]:

- Created multi-threaded program using C and POSIX API, emulated the process of token bucket with the statistical summary, with the concern of mutual exclusive and deadlock.
- Implemented a Linux similar operating system, Weenix in C programming language. Completed a UNIX-like multi-threaded/ processed kernel with Virt*ual File-System(VFS)* layer and a *Virtual Memory (VM)* layer in a team of 4