PRAKHAR SHARMA

Phone: +65-9370-9378 | Email: prakhar.gtm@gmail.com
Personal Website: https://sharmaprakhar.github.io

RESEARCH EXPERIENCE (National University of Singapore) – Research Engineer (10 months)

- Hardware architecture design for Superpixel Segmentation computer vision algorithm
- Hardware architecture design for 2D convolution based Gaussian filter for image convolution for custom pixel throughput and latency
- Hardware architecture design for Implementation of Union-Find and Minimum spanning tree (graph) algorithms
- Custom IP generation and integration on Xilinx Zynq-7000 FPGAs (HW-SW co-design)
- Bayesian Networks (machine learning) implementation on MATLAB and SAMIAM
- Exploring the Machine learning and Deep learning paradigm for accelerator implementation on chip and on FPGAs
- Variability Analysis in VLSI circuits and reduction in Monte Carlo runtime for yield analysis – an importance sampling approach
- Color Feature extraction using OpenCV in C++
- RTL2GDSII schematic to tape out experience using Cadence and Calibre design tools

PROFESSIONAL EXPERIENCE – Advanced Micro Devices (AMD – 2.5 years)

- Skew specs based Clock-Spine design for discreet GPUs – floorplan based clock routing and tree synthesis (14nm)
- Clock buffer design with optimized drive strength and minimum power delay product (released as a Std.Cell. Library)
- Automated Layout and Routing generation using Perl, tcl, Cadence-Skill and Python
- CAD toolchain regression maintenance for design tech-file consistency
- ADM in memory SRAM bit-cell and Most Probable vector analysis of worst case degradation
- Quasi Monte-Carlo Importance Sampling implemented in yield analysis for Flip-Flops

Software Proficiency: Python, C++, Matlab, Verilog, Unix, Perl, Tcl, Skill, Embedded C, Labview (chip testing)
CAD Tools: Cadence, Synopsys, Mentor graphics, Xilinx – Vivado and SDK, Caffe (deep learning)
Relevant coursework: Analog and Digital VLSI design, Advanced Computer Architecture, Microelectronics, Machine Learning, Deep Learning

PUBLICATIONS

- Estimation of SNM in Latches and subsequent formation of a 10-T CNFET bitcell **[IEEE - WISES 2013 Pilsen Czech Republic]**
- Method for Sizing Complex CNFET Bit cells for Balanced Read Write Operation **[IEEE – 6th ICCCNT Texas, USA]**

MOOC courses

Machine Learning (Stanford)
Probabilistic Graphical Models(Stanford)
Advanced Computer Architecture (Princeton)

RELEVANT PROJECT DETAILS

Hardware architecture design for Superpixel Segmentation (NUS, 2016)
Segmenting an image in to semantically important sections has been demonstrated to be crucial for image classification and analysis. We explored the hardware implementation of the State of the art graph based implementation by P. Felzenszwalb(MIT) and D. Huttenlocher (Cornell). Custom Architecture for 2D convolution based Gaussian filter, Union-Find and Minimum spanning tree algorithms were designed which reduced the memory access patterns and overall complexity of the original implementation. The output format of this implementation is feature extraction friendly, which is achieved by running through a trained Bayesian Network. The work to map the network and the feature extractors is currently in progress.

Custom IP generation and integration on Xilinx Zynq-7000 FPGAs (HW-SW co-design)
Xilinx Zynq devices are the most advanced FPGA devices available today. I was responsible for self-learning and preparing a knowledge base of techniques involved in generating a custom IP and creating an embedded design with the Xilinx tool chain and carrying out debugging and porting to the hardware. My Git repository on zynq devices offers open-source ready to deploy DRAM access systems to be integrated with custom user logic.

Exploring the Machine learning and Deep learning algorithms for on chip and FPGA accelerator implementation
Deep learning offers the best accuracy among all vision algorithms, though it is impractical to implement it on chip due to large parameter storage requirements and amount of compute logic involved. This project is a literature survey of the state of the art and comparable algorithms and exploration for improvements (while maintaining reasonable accuracy) to make them hardware (ASIC) friendly.

Variability analysis in VLSI circuits and reduction in Monte Carlo runtime for yield analysis (AMD, 2014)
Traditional Monte-Carlo method for flip-flop stability is slow and inaccurate. The SNM calculation method was shifter from butterfly curve to N-curve method. Quasi Monte-Carlo using Latin Hypercube Sampling and Sobol sequences were implemented first to reduce run time in multidimensional variable parameter space and an observable run time reduction was achieved using HSPICE. Importance Sampling for one variable parameter in the device models for FinFETs was implemented and the current research is going on to extend this methodology to real industry design models with multidimensional variable parameter space

Method for Sizing Complex CNFET Bitcells for Balanced Read Write Operation
A new experimental method to size CNFET circuits, especially SRAM bitcells is outlined. The trade-off philosophy between read and write operations in sizing is established. The transistors in a complex 10T SRAM bitcell are sized using the algorithm and data is presented to verify the balancing criteria and balancing point. The results are replicated in a simpler 6T bitcell using the same algorithm and similar analysis is carried out. The algorithm achieves balance in the SRAM bitcell between the read and write trade-offs whilst maintaining good individual read and write metrics. Similar work is done on CMOSes to corroborate the algorithm's universalness

Clock design for discreet GPUs (AMD, 2015)
AMD designs cutting edge APUs, CPUs and GPUs and the clock spine design of these products was the responsibility of the team I worked with. We designed drive buffers for clock routing from the PLL to the logic blocks. The first phase was to carry out skew –match simulations for technology specific buffer drive strength. The second phase was to design buffers schematic and layouts and release them as standard cell library. The third phase was to lay down clock trees and carry out CTS (clock tree synthesis) on logic blocks. The final phase was to verify the design against specs.