Jatin bhateja

05072 ATS Advantage, Plot No-17, Ahinsa Khand-1, Indirapuram| Ghzaiabad, Uttar Pradesh, 201014 | 8826368490| [jatin.bhateja@gmail.com](mailto:jatin.bhateja@gmail.com)

# Objective

Seasoned software engineer having 20+ years of rich hands-on experience in Compilers and EDA front-ends, seeking a challenging position in the area of cutting-edge technologies where my conceptual and academic knowledge will be used effectively to create a differentiated product and an integrated solution.

# Skills & Abilities

Languages and Tools:  C, C++11, Python, Tcl, Java (Core), Bash, Assembly language (x86). Flex, Bison, ANTLR. Verilog, VHDL, Git, Perforce, CMake.

# Experience

## Cloud Software Development Engineer / Sr. Staff Engineer, Intel Corporation **Dec’18 – Present**

### Remote Employee, Bengaluru

* Part of the Java platform optimization team since joining in December 2018. Currently focusing on adding low-precision type support in Java, making it future-ready for next-generation AI workloads and Vector Databases. Also spearheading Diamond Rapids software enabling efforts in JVM, closing the competitive optimization gaps against ARM servers, and Vector API – Valhalla integration efforts, working in close collaboration with Oracle teams.

Following is a gist of my year-wise significant contributions to OpenJDK since joining Intel:

* **2019**: C2 compile Generic Operands Support
* Significant infrastructure changes in instruction selection, enabling the merging of lots of instruction selection patterns. This resolved one of the major bottlenecks in Vector API 1st Incubation acceptance into the mainline OpenJDK. ARM later adopted this solution to unify their Neon and SVE backends.
* **2020**: AVX512 differentiating optimizations
* Designed and implemented an advanced Ternary Logic optimization that folds logic cones into one 3 input LUT to generate ternary logic instructions introduced by AVX512 ISA.
* Implemented partial inlining of small copy/mismatch methods using AVX512, showing 2-3x speedup.
* Optimized stubs for copy, fill, and mismatch methods routinely used in almost all Java workloads.
* Optimizations of various math operations like ceil, floor, round, floating point min/max, signum, float to integral conversion.
* **2021**: Implementation of Java Enhancement Proposal (JEP) 417: Vector API (Third Incubator)

Predicated operations form the backbone of data parallel programming and help in translating conditional logic into straight line data flow. Vector API third incubator added AVX512 optimizations for predicated/masked operations enabling users to write efficient data parallel code in Java.

* Jatin was leading contributor for predicated vector operations support resulting into around 2x performance improvement on AVX512 targets. He extended register allocator to handle opmask registers and added support for masked AVX512 instruction code generation. He also improved the performance of routinely used masking operations, firstTrue, lastTrue, anyTrue, allTrue.
* **2022**: JEP 426: Vector API (Fourth Incubator)

Jatin spearheaded the design and development of the JEP in collaboration with OpenJDK community partners (Oracle/ ARM).

* Optimized implementation of various bit level operations, BIT\_COUNT, LEADING\_ZERO\_COUNT, TRAILING\_ZERO\_COUNT, REVERSE, REVERSE\_BYTES.
* Added powerful vector compress / expand APIs leveraging AVX512 instructions. These APIs target optimizing columnar data-base predicate pushdown and was used by OpenSourceDB team to optimize ORC reader in prestoDB open-source showing 1.5x gains.
* **2023**:

Valhalla – Vector API Integration

OpenJDK project Valhalla led by Oracle introduces value types in Java and is slated to be integrated in mainline OpenJDK in one of the upcoming releases. Java Vector API vectors are immutable quantities and so should be implemented as value types. This is a key requirement for Vector API incubation exit.

* Jatin is leading the Vector API incubation exit and introduction as a preview/product feature in the OpenJDK community. He is the only contributor on this effort from Intel. He came up to speed on Valhalla specification and its complex implementation. He then designed and implemented a prototype of Java Vector API over Valhalla working closely with ARM engineers. As part of this he identified important gaps in Valhalla design/implementation and shared his findings with Oracle JVM architects.

Intel CESG PRC team collaboration

* Delivered optimized parquet unpacking algorithm, which was upstreamed into parquet-mr mainline and received several testimonials from customer Alibaba.
* Collaborated for 1.5 years with PRC team on Vector API backporting efforts to Alibaba JDK Dragonwell 11.
* **2024**:
* APX support for DMR: Jatin kicked off OpenJDK DMR SW enabling effort with APX EGPR register allocation support, extended GPR state save-restoration using efficient PUSHP2/POPP2 instruction, and new setZUCC ISA support. He also reviewed other team members work on APX encoding and code generation support in the Java runtime. Jatin setup end-to-end validation flow using SDE for OpenJDK. He discovered, root-caused, and reported multiple blocker issues to SDE team.
* Float 16 support in Java: Jatin took initiative and collaborated with Oracle/ARM to add Float16 type support as an incubating feature in JDK 24. He then followed it up with optimized scalar float16 operations using Intel FP16 ISA which is integrated into OpenJDK towards JDK 25. Auto-vectorization support is in progress.
* **2025**
* Jatin is working on identifying and closing the performance gaps with ARM AARCH64 servers (Graviton).
* Jatin has started a feasibility study on adopting MicroTSX for large atomic field updates in Valhalla.

Jatin is the only Intel OpenJDK development team member outside US and has made significant contributions to enable Intel's differentiating features in OpenJDK.

Following are links of my open-source contributions from Intel towards various OpenJDK projects [1][2][3], he also collaborated with CESG PRC team to optimize parquet-mr reader using Java Vector API [4]

Prior to joining Intel, Jatin worked in open source for a few years and made contributions to LLVM Compiler [5] and developed JVM backend from scratch [6].

[1] <https://github.com/search?q=repo%3Aopenjdk%2Fjdk+author%3Ajatin-bhateja&type=pullrequests&ref=advsearch>

[2] <https://github.com/search?q=repo%3Aopenjdk%2Fpanama-vector+author%3Ajatin-bhateja&type=pullrequests&ref=advsearch>

[3] <https://github.com/search?q=repo%3Aopenjdk%2Fvalhalla+author%3Ajatin-bhateja&type=pullrequests&ref=advsearch&p=3>

[4] <https://github.com/apache/parquet-java/pulls?q=is%3Apr+is%3Aclosed+VectorAPI>

[5] <https://github.com/search?q=repo%3Allvm%2Fllvm-project+author%3Ajbhateja&type=commits&ref=advsearch>

[6] <https://github.com/jbhateja/llvm_jvm/tree/master/lib/Target/JVM>

## Senior Technical Lead, Gemalto NV **Dec’17 – Nov’18**

### Noida, Uttar Pradesh

* Gemalto is a world leader in Security solutions; Sentinel LDK is their flagship product for Software Monetization which applies multi-layered security over application targeting various platforms and operating systems.
* Encryption, Obfuscation, load time randomization, licensing and code virtualization are various ways in which application is secured.
* Code virtualization is a method by which user specified functions (IP) are converted to VM assembly and run over virtual machine in TEE (trusted execution environment) of a hardware dongle / Java Card.
* I'm involved in developing a solution from scratch which takes C/C++ source as input, auto selects function which can be virtualized and then generates JVM byte code for them.
* Scope of my work involves making custom transformation over IR generated by Clang (a C/C++ frontend) in order to make IR consumable for backend processing.
* Developing a new backend for LLVM targeting JavaVM, this involved writing SelectionDAG based instruction selector, various tablegen files (.td), custom MI SSA passes, custom AsmPrinter for JVM which generates output ASM in format acceptable by Jasmin Assembler, integrating Jasmin assembler for compiling ASM to .class file.
* Have been able to successfully deliver PoC for new JVM backend, composite support, casting (reinterpret/expand/narrowing), support all integer and long operations as per the Java Card VM specification 3.0.5.

## Open Source Contributor, LLVM Foundation **Feb’17 – Present**

### Noida, Uttar Pradesh

* LLVM umbrella covers a set of complete tool chain for compiling HLL like C/C++ targeting various architectures.
* It includes clang (a C/C++ unified frontend), opt (target independent optimizer), llc (backend), llvm-mc (assembler), lld (linker) and lldb (debugger).
* I have good understanding of X86 ISA and various Intel Microarchitectures.
* Have deeply explored and contributed mainly to backend for two targets X86 and WebAssembly.
* Have good hands on over backend (llc) and optimizer (opt) especially in following areas:-.
  + Instruction selection and scheduling.
  + Table generation based target file.
  + IR level transformations and analysis passes.
  + Scalar and Vector transformations.
* Contributions and patches under review can be visited at <https://reviews.llvm.org/p/jbhateja/>

## Lead Member Consulting Staff, Mentor Graphics Corp. **Aug’12 –Jan’17**

### Noida, Uttar Pradesh

* Worked in following areas at Mentor Graphics :-
* **FPGA Prototyping :**
  + Worked for a short time over a new yet to be announced offering from Mentor Graphics in area of FPGA based ASIC prototyping based on Flexras Flagship timing driven partitioner.
  + Gained good understanding of complete end-to-end flow of the compile side i.e. netlist reading, resource estimation, design modifications needed in ASIC netlist and partitioner.
  + Partitioned design is then pushed through Xilinx Vivado PnR flow to generate bits to be loaded on FPGAs.
* **Questa Simulator:** My work in Questa has been mainly into simulation kernel and elaboration.
  + Following is a gist of my Simulation projects:-.
  + ***Fine grained parallism for vsim kernel:***
    - Research and prototyping for parallel simulation based on multi-threading.
    - Prototyped a PoC by parallizing VHDL simulation cycle at process level, this involved dynamic load distribution of processes and active signals between various threads, honoring the dependency b/w signal evaluations and synchronizing the threads at appropriate sync points in the simulation cycle.
    - Was able to achieve the thread/task parallelism and it showed nearly liner speed-up in simulation time till 4 threads on synthetic and some real world designs.
  + ***VHDL performance optimizations for FPGA based designs***, specifically Xilinx (both Vivado and ISE).
    - Implemented from scratch several targeted optimizations like LUT Optimization, Clock suppression for BLOCK RAMs, Distributed RAMs, SRLs and Flops (FD.
    - Delta re-timing to for several unisim primitives and sparse memory optimization.
    - Significant performance improvements ranging from 1.5x-10x were seen in customer and open source designs due to this effort.
  + ***NextGen ADSM solution:***
    - Implemented SPICE elaborator, this involved extending Questa's multi-phase elaboration for SPICE on the lines of Verilog elaboration.
    - Research and prototyping for SPICE interconnect analysis and resolution needed for boundary element insertion.
    - Effort was split between Grenoble and Noida.
    - Traveled to Grenoble in Dec'13.
  + Has been among top 10% of workforce, got several stock grant awards for being top contributor.

## Senior Software Engineer – II, Xilinx Inc. **Aug’10 –Aug’12**

### Hyderabad, Andhra Pradesh

* Following is gist of my projects:
* ***Xilinx Synthesis Technology (XST)***
  + XST is Xilinx's flagship FPGA synthesis tool, it accepts Verilog/VHDL/EDIF files as input and generates synthesized netlist targeted to various Xilinx FPGAs (virtex5/6/7, spartan3/6, kintex7, Artex7 etc.).
  + Owned BRAM inference flow of XST.
  + Owned map -global\_opt flow in ISE product, which used ABC logic optimizer to perform powerful combinational and sequential synthesis optimization over unisim mapped netlist.
  + Overall quality improvement in tool by fixing lots of customer design halted issues, logic incorrectness and QoR related issues reported time to time during customer evaluations.
  + Was one of the founding members of team at Hyderabad, traveled to Grenoble twice for technical discussions and KT.
* ***Vivado Synthesis (Rodin Synthesis)***
  + Next Generation synthesis tool by Xilinx, supports Virtex7 and higher families.
  + Best in class synthesis tool, leverages path breaking chip level synthesis technology of Oasys Design Automation
  + Worked into DFG level optimization and custom netlist generation for targeted patterns.
  + Travelled to US (San Jose) for 1 month for getting trained on new tool initially.

## Lead Member Technical Staff, Mentor Graphics Corp. **Aug’06 –Aug’10**

### Noida, Uttar Pradesh

* Following is gist of my projects:
* ***HDL Compiler Frontend :***
  + Worked over Cheetah and MVV analyzers used for Verilog and mixed language lexical analysis, parsing, semantic analysis and elaboration.
  + Supporting various parsing modes like fast fault tolerant mode, complete mode for full parsing and semantic analysis.
  + Different modes of parsing catered to different clients like design editor , RTL compiler etc.
* ***Editing Engine:***
  + Developed a design refactoring engine for Verilog, which supported various design manipulation operations like module level and hierarchical port additions, outlining, instance moving across hierarchy.
  + This tool worked in conjunction with DRC (design rule checker) and performed auto-corrections over design for violations generated by rule-checker.
  + A unique format preserving decompiler generated the modified Verilog file which preserved user's comments from original source code.
* ***Veloce Emulator Visulizer:***
  + Was part of visualizer team for a short time, got opportunity to work over various widgets of visualizer like signal browser, hierarchy browser.
  + Performed optimizations by moving performance intensive code from TCL to C for quick loading and bring-ups widgets contents.

## Software Engineer, Conexant Systems **Dec’05 –Aug’06**

### Noida, Uttar Pradesh

* **Compilers and Tools team member, following is gist of my project:**
  + Worked over an open source C Compiler SDCC (small device C compiler) which targets micro-controllers.
  + Project was about targeting SDCC to one of Conexant's DSP chip set named Cougar.
  + Worked in various phases of compiler that is front-end, semantic analyzer, ICode (intermediate code generation), optimizer and back-end.
  + Implemented "Direct Addressing Pass" of compiler which produced the base + offset combination for all the non-scalars references.
  + Implemented "String literal optimization Pass" of compiler which saved significant memory in read only area.
  + Designed and implemented "Global Initialization Pass" of compiler.
  + Overall quality improvement and stabilization of compiler.

## Subject Matter Expert, Amdocs **Sep’05 –Dec’05**

### Pune, Maharashtra

* ***AFG - RBMS (Rule based management system).***
  + Worked for a small duration into RBMS. It’s a tool which takes as input rules (governed by the business) in English like dummy language and it produced the equivalent C code.
  + C code produced by this tool is used for performing various functions like editing, validation, determination and mapping of various input formats.
  + RBMS is a module of Acquisition Formatting and Guiding phase of the core billing product of AMDOCS (Ensemble and Charging).

## Member Technical Staff, HCL Technologies **Aug’04 –Sep’05**

### Noida, Uttar Pradesh

* ***System Software Group, Fortran-90 SX compiler (NEC).***
  + Worked into Intermediate code generation of the FORTRAN compiler targeting super computer (SX).
  + Involved generation of intermediate language from parse tree and applying various optimizations henceforth.
  + Different optimizations at IR level involved fusion analysis, work array reductions, inline expansion etc.
  + Wrote an optimized inline expander for RESHAPE intrinsic function of FORTRAN.
  + Scalar optimization of ELEMENTAL Intrinsic functions when they contain derived type pointer argument.
  + Stabilization of run time array subscript checking mechanism of compiler.
  + Optimization of TANH intrinsic function of FORTENA95. Function was written in SX-4 assembly language.
  + Received "Certificate of Excellence" in Dec 2004.
  + Studied the Front end of GNU Gfortran compiler for supporting Fortran F2k constructs.
  + Worked on developing a preprocessor, which translates the FORTRAN 2003 programs to FORTRAN 95 using ANTLR.
  + Automation of test suite generation and test suite review.
    - Generated Scripts to automate the process of test suite review and automatic rectification of faulty test cases.
    - Developed a script using Perl, which automatically generates the FORTRAN program.

# Education

## B. Tech (Computer Science and Engineering) **Jul’00 –Jun’04**

## Maharaja Agrasen Institute Of Technology,

## GGS Indraprastha University, New Delhi

* Graduated in Computer Science with a distinction and overall CGPA of 81%.
* Worked over following projects during graduation:
  + ***Compiler Construction :***
    - As a final year project developed a small compiler for a dummy language in C.
    - Hand written lexer and parser (top down LL(1)).
    - Basic optimizations over parse tree.
    - Rudimentary backed which translated parse tree into an 8086 assembly.
  + ***Implementation of STOP and WAIT ARQ protocol***
    - As a pre-final year project developed a client side and server side application which communicated using above mentioned protocol.
    - Protocol was developed using Message Queues in C on Linux.
    - Techniques like re-transmission on timeout, piggybacking & removal of duplicate messages were handled.
* Secured 91.71 %ile in GATE 2004.
* Secured 94%ile in Competence in Software Technology (CST) G Level conducted by C-DAC (M&B) in 2004

## Post Graduate Certificate in Machine Learning and Deep Learning **Dec’19 –Jun’20**

**International Institute of Information Technology (IIIT), Bengaluru**

# HOBBY PROJECTS

JLang Compiler – Small compiler for educational purpose.

* This project was made with an intention to understand the compiler working at a deeper level right form the lexical analysis till code generation.
* On compiling the source program written in a small dummy language named ***Jlang*** user is able to see the output produced by each phase of compiler. This helps the user to understand the concept behind each phase and gives an idea about the working of various phases.
* Optimizer comprises of optimizations like *constant folding*, *algebraic optimizations*, *CSE*, *constant/copy propagation* and *dead code elimination*.
* Project is hoisted at **GNU Savannah** for benefit of larger audience.

<http://savannah.gnu.org/task/?6484>

# INTERESTS

* Programming in "C/C++", Data structures, Algorithm design and analysis, Compilers, Operating System, Microprocessor, Computer Architecture, Digital synthesis and Simulation.