Senior Validation Engineer, Python Automation Senior Validation Engineer, Python Automation Software Engineer Hillsboro, OR Aspiring to continue my career creating and operating system software, tools and applications in Linux, Windows, other operating systems, or embedded/firmware. Eager to work as a contractor, full-time or part-time, employee. I hope for a mutually beneficial work environment wherein the depth of my skills is useful and valued. Very knowledgeable of PC, IA32 and EM64T CPU (micro-processors), chipsets and systems using them; other CPUs and micro-controller use also. Highly proficient in Linux/Unix/C/ASM/C++ environment; good in Windows too. Broad understanding of software engineering, refined by decades of challenges across diverse projects. Strong work ethic, taking unparalleled pride in my work, consistently delivering excellence. I assume responsibility and tenaciously drive toward project goals. I m known for being well organized in everything I am involved in. Authorized to work in the US for any employer Work Experience Senior Validation Engineer, Python Automation Intel - Hillsboro, OR May 2018 to December 2018 Developed, executed and debugged FPGA validation system on Linux. Infrastructure and tests were Python with some large Bash scripts. Built and configured Linux systems; installed Linux OS, drivers and packages. Debugged failures related to OS, drivers, automation and network. 1. Developed DLA (Deep Learning Algorithm) regression testing and installer. 2. Wrote validation test specifications. 3. Created PCI error and other HW error detection run parallel with all tests. 4. Dramatically improved dev, triage and debug by instrumenting automation error exits. 5. Standardized log file and console output similarly. 6. Foresaw a tool to install automation, drivers and configure systems for FPGA execution. 7. Envisioned test execution supervisor and a regression analysis tool to gather and summarize results. 8. Translated large, primary regression tests, Bash to Python. Wrote a tool achieving 80% of this process. Test & Tools Capabilities SW Engineer/Senior Developer, Advanced CPU March 2014 to June 2016 Expanded an LLVM compiler and linking locator, Linux and Windows hosted, in C++ and assembly. Resulting tests executed on hardware, emulators and simulators. Much target info was available at compile time, so references by class, generated as immediate values; successive value uses were deeply compiler optimized. For PCI device config. register access, no register read was performed, the

value was available at compile time. This system drastically reduced test execution time and size. 1. Expert owner of linker and memory allocation across all address spaces, developing hardware-specific detection and management of diverse memory and MMIO regions used by framework and tests. 2. Made numerous paging changes including introduction of non-identity mapped support. 3. Dramatically improved dev/debug by adding DWARF, source-level (symbolic) debug data, for source-level debug. Rewrote object writer modules, with inheritance and derived classes, writing data into ELF32/64 files. Complete ownership required research, planning, initiating beta deployment, and coordination across organizations. Leveraged capabilities by engaging groups that create or test DWARF. Defined QA testing to leverage Linux SDT extensively. Result was maximum confidence in file format accuracy. 4. Produced initial VT-d user (language) interface and linker VT-d paging support. 5. Added framework for all special memory regions, e.g.: PRMRR, graphics aperture, cache slice, concurrency, etc. 6. Provided test-card use, generalizing detection, config and grammar for memory thereon. 7. Eliminated risk of linker overlaps from complex resource interactions and linkage to other tool outputs. System Software/Tool Engineer/Architect/Tech Lead/Team Lead 1992 to 2016 More than 25 years as a software engineer and: ? Developer, 25 yr ? Debugger, 13 yr ? Technical Lead, 6 yr ? Team Lead, 7 yr ? SW Architect, 8 yr ? Performance Architect, 4 yr ? Designer, 25 yr ? Project Planner, 7 yr ? Requirement/issue management, 6 yr ? Meeting chairman, 7 hr. ? Cross-site facilitator, 7 yr ? Cross-org. coordinator, 6yr ? Senior System SW Eng., 25 yr ? Senior Tool Eng., 25 yr Conceived and built software and tools for external and internal customers. Led teams and technical lead of CPU simulators and large systems, vital to designing and qualifying CPUs and chipsets. Forged processes to ensure system operation. Improved product execution speed, adding perf. improvements and analyzing perf. problems, as a performance architect. An innovative initiator that drove toward results with solid customer orientation. Believe that software dev includes creating tools that empower product development, for group or individual compute environments. I thoroughly enjoy writing code, with excellent software dev methods which create code others can read and understand. Component Design Eng./Senior Developer May 2011 to March 2014 Coded for Linux,

mainly w/ drivers to detected device config and controlled devices via register use, memory mapping, interrupt handling and power transitions, like PEG C7 (via ACPI). 7. Efficiently supported new CPU products, using common source for each device, so driver binaries ran on any system. 8. Became lead infrastructure developer of new desktop CPUs; owned eight drivers: CPU core, un-core, audio, etc. 9. Supported diverse buses in un-core (south cluster) driver: PEG, DMI, PCU, interrupts, IOAPIC, GPIO, PMC etc. 10. Economically enabled successive CPU and device family testing, with powerful automation to process large, complicated RTL source trees, generating driver register data lists, across diverse, per device, memory/IO regions. 11. Comprehensive RTL capture for drivers, typically pre-empted requests for new micro-architectural support. 12. Key enabler of CPU validation and debug, unblocking model and early silicon trouble, throughout the environment. 13. Chosen member of silicon power-on team for multiple CPUs. 14. Able to debug 40 pre-sighting and 2 silicon bugs, on one CPU product. Solved problems in EDRAM RTL fuse production; then instigated discussions to improve RTL tools and complete EDRAM unit and collateral. 15. Awarded several times, including a division (-1) award, for enabling graphics and power-management testing during first, Broxton CPU power-on. Brought execution online another time, deciding 'Ed' worked for config file editing. 16. Expert for booting scores of releases on countless models; debugging Linux or tool problems. 17. Accelerated model availability for pre-silicon groups, defining methods, to specify absent registers. needed for boot. 18. Solved unavailable CPUID/device IDs in early/broken models, adding ID spoofing into central library APIs. 19. Defined and created functional ram-disk and EMMC images and working simulator/OS/driver recipes. 20. Sped driver mounts from >3 hours to tens of minutes, for crucial simulations, with OS config changes. 21. Enabler of infrastructure so actual test runs could begin; providing initial device driver system mounts. 22. Un-blocked 10 key validation areas, devising new architectural access interface, eventually used in a CPU BIOS. 23. Saved several scores of developer hours, with tool to detect unexpected CPUID/device ID info. Architect/Tech Lead/Team Lead/Component Design Eng New CPU August 2009 to May 2011 1. Defined strategic objectives and key contributor to creating a unique identity for system #2. Architect/Tech Lead/Team Lead/Component Design Eng Major CPU May 2005 to August 2009 2.

Delivered unprecedented stability of system libraries, tests and infrastructure, as Team Lead (system #1). Senior Software Eng., Major CPU validation system #1 Major CPU January 2004 to May 2005 Component Design Engineer CPU Architecture June 2002 to January 2004 BIOS Engineer, System Validation BIOS Intel June 2001 to June 2002 CPU Architecture Performance Engineer, Architecture performance CPU tools Intel October 1995 to June 2001 3. Expert in CPU/system performance, e.g. micro-arch, memory/IO, caching, multi-threading (MP). 4. Devised most of the complex micro-code, e.g. most EM64T, privileged, segmentation, paging, VM/FP assist, snoops, SMC recovery, interrupt/exception/fault and resume. 5. Contributions were visible to high level organizations beyond CPU architecture. Team Lead/Senior Software Design Eng., Pentium-Pro Chipset System Validation Intel April 1995 to October 1995 Team Lead/Tech Lead/Senior Software Design Eng March 1995 to March 1995 6. Led pivotal Pentium-Pro CPU/system performance model for early years, internal/external design. 7. Owned memory, caching, cache coherence and external bus models. 8. Converged perf to within 10% of RTL on 90% of required workloads. Model tuned an application from 68 seconds to about 6; on RTL it was 6.02. Software Design Engineer, Feats Intel 1995 to 1995 9. Invented 11 comm. systems (one Ethernet) for: Sdm960Mc, Sdm960Kb, Icd960Ca, Ice486, Pentium, etc. 10. Delivered benchmark data for CPU and floating-point units, used in Intel marketing claims. Education Computer Programming/Software Technology in Software Development Portland Community College - Portland, OR 1983 High school in Advanced Math/Science Ashland High School Skills Architecture (5 years), BIOS. (3 years), debug (5 years), Linux. (5 years), segmentation (5 years) Certifications/Licenses NCRC Platinum Certificate April 2018 to Present Additional Information Skills Very knowledgeable of PC, IA32 and EM64T CPU (micro-processors), chipsets and systems using them; other CPUs and micro-controller use also. Highly proficient in Linux/Unix/C/Assembly code/C++ environment; good in Windows too. Have broad understanding of software engineering, refined by decades of challenges across diverse projects. Strong work ethic, taking unparalleled pride in my work, consistently delivering excellence. I assume responsibility and tenaciously drive toward project goals. I'm known for being well organized in everything I am involved in. Developed

numerous types of code: ? Device driver, 9 yr, 26 times ? Operating system, 6 yr, 6 times ? BIOS or Boot code, 10 yr ? Embedded code, 7 yr, 9 times ? Hardware control code, 13 yr ? CPU/system simulators and FPGA emulators, 9 yr, 5 times ? Automation tools, 7 yr ? Multi-thread (MP) control code, 14 yr ? Communication systems, 5 yr, 11 times ? Loadable Linux kernel modules, 3 yr ? Linux debug, 4 yr ? CPU/system memory/cache/coherency, 4 yr ? Multiple host control, 1 yr, 2 times ? Binary translator, 1 yr, 1 time ? Host-target code 11 yr ? Self-hosted code 14 yr ? CPU micro-architecture/system performance, 9 yr ? Debugger, 4 yr, 4 times ? Compiler, 3 yr, 2 times ? Linker/locator, 4 yr, 3 times ? Assembler, 2 yr, 2 times ? Interpreter, 1 yr, 1 time ? Paging/segmentation, 4 yr ? Interrupt/exception/fault, 5 yr ? PCI/PCIe, USB, SATA, RS-232, RS-422, PMRR, EPROM, SRAM, flash memory, 8 yr ? VMX, 2 yr ? VT-d, 1 yr ? EPT, 1 yr ? SMM, 1 yr ? Floating-point, 3 yr ? Memory Management, 6 yr ? Object file processors, 3 yr, 4 times ? Data analysis, 2 yr ? Graphics, 1 yr ? Developer tools, 6 yr ? Internet TCP/UDP/IP, 1y Expert use of compiler, assembler, interpreter and script languages: ? Python, 2 yr ? C, 25 yr ? C++, 11 yr ? Assembly code (MS/Linux/Intel), 15 yr ? Perl, 11 yr ? Shell scripts SH/Bash/Csh/Tcsh, 11 yr ? LLVM, 3 yr ? YACC and Lex, 1 yr ? Make, 2 yr ? XML, 3 yr ? DOS/cmd batch script, 15 yr ? Fortran, 1 yr ? Basic, 6 mo. ? Cobol, 3 mo. ? PLM-86, 2 yr ? RTL, 6 mo. ? BFL, 3 mo. ? Tickle, 1 yr Coded for many operating systems (OS), embedded and host/target environments; within many OSs: ? Linux/Unix: RHEL, Ubuntu, CentOS, BSD, 25 yr ? Windows, 20 yr ? MS-DOS, 25 yr ? IBM OS JCL, 1 yr ? VMS, 1 yr ? RMX-86 RTOS, 1 yr ? ISIS, 3 yr ? CPM-86, 2 yr Collaborated dev within these environments: ? Git, Gerrit, Artifactory, 8 yr ? Agile/Scrum, 3 yr ? Continuous integration, 7 yr ? MS Office, Skype, Outlook, Skype, 6yr ? Code Collaborator, 7 yr ? MS Project, 6 yr ? Instant-messaging, 7 yr ? Skype, 4 yr ? Lync, 7 yr(Effective usage of developer, debug and computing tools: ? DDD, XDB, GDB, Eclipse, Visual Studios, Linux SDT language toolkits, Vi/Vim ? Hardware tools: ITP, Lauterbach/Trace32, logic-analyzers, oscilloscopes ? MS Office: Excel, Outlook, (MS)Project, Word ? Source repository and version control: Git/SVN/CVS/RCS/PVCS, bug trackers, mailers Copious work with/within various object file formats: ? ELF32, ELF64, symbolic and symbolic debug (e.g. DWARF), COFF,

PE, OMF, Intel HEX (I*HEX)

Name: Walter Carr

Email: kgreen@example.org

Phone: (705)433-9553