CV for Wenxue Xie (Cory)

CONTACT INFORMATION

Name:	Wenxue Xie (Cory)
Phone:	13581787949
Email:	cory.xie@gmail.com
Github:	www.github.com/CoryXie
Blogs:	www.cnblogs.com/CoryXie

EDUCATION

- B.S. in Communication Engineering, Beijing Information Technology Institute, 2001.9~2005.7
- Languages: Chinese (native), English (fluent, College English Test Band 6)
- A continued self-training on Computer Science (especially Computer Architecture)

FORWORDS

"Again, you can't connect the dots looking forward; you can only connect them looking backwards. So you have to trust that the dots will somehow connect in your future." -- Steve Jobs. The following lists will tell a story of how my dots connected.

AWARDS AND HONORS

- **Awards Received During School**

  - 2000 Received title of "Excellent student cadre of Sichuan Province" during high school due to my contribution as President of the Student Union.
  - 2002 Received 1st prize of China's first "National Merit Scholarship" and went to the Great Hall of the People to accept the award (I was one of 3 students to receive the award in my college).
  - 2004 Received 3rd prize of "The National Undergraduate Electronic Design Contest-Embedded System Design Invitational Contest" with a 3 person development team.
  - 2004 Technical paper "JINI and Java Spaces based LAN messaging system" published in the 7th annual "National Java Conference" by Sun Microsystems.
  - 2004 Received 2nd prize in "BITI Students' Science and Technology Festival" with my "JINI and Java Mail based messaging system".
  - 2004 Received the "<China Computer World> Magazine Scholarship", which was only for some selected colleges and two candidates per college every year.
  - 2001~2005 Received "Outstanding Student Scholarship" each semester, twice for 1st prize, twice for 2nd prize, and 3 times for 3rd prize.

- **Awards Received During Career**

  - 2009 Promoted to be Senior Engineer from Engineer at Wind River Systems.
  - 2010 Promoted to be Member of Technical Stuff from Senior Engineer at Wind River Systems.
  - 2010 Received "Business Focused and Customer Driven Award" at Wind River Systems due to my contribution to various customer support efforts.
  - 2013 Received "Optimistic, Upbeat, and Passionate Award" at Wind River Systems due to my contribution to a high value customer support effort.

PROFESSIONAL HISTORY

- **2011.01-current, Member of Technical Stuff, China Development Center of Wind River Systems**

Due to my outstanding performance to the USB product line, I was promoted to be Member of Technical Stuff in December 2010, 1.5 years after working as Senior Engineer with Wind River Systems. Beyond working as the technical lead in the USB team for product maintenance and feature enhancements, I expanded my career path to work as a "full stack developer" in the field of System Software. In this role, I was responsible for designing and implementing big new features that satisfy customer requirements. The following are some new features I have created or been working on for VxWorks.

  - **Fully Featured USB 3.0 Stack**: As USB 3.0 has becoming mass production delivery on most new platforms, supporting USB 3.0 on VxWorks became an important requirement. I took the responsibility to design and implement a new USB 3.0 stack, which included the following features: 1) A set of new backward compatible USBD APIs to support USB 3.0 Dual Bus Architecture; 2) A new USB 3.0 host controller driver (xHCD) supporting all major features of xHCI specification, tested with all available xHCI controller chips at the time; 3) A set of APIs supporting USB 3.0 new capabilities (e.g, Bulk Streams); 4) An enhanced USB Mass Storage Class driver supporting UAS/UASP with USB 3.0 Bulk Streams (a new supporting module based on the GEN2 MSC class driver created in 2008). The USB 3.0 stack has been delivered as a fully supported Wind River product. The raw performance was tested to be outperforming other mainstream Operating Systems. In doing this work, I also participated in discussions with NA team and made key idea contribution for improving the PCI/PCIe MSI/MSIx support mechanism in VxWorks.

  - **Pluggable I/O Scheduler Framework**: There is a thin layer called XBD (Extended Block Device) layer sitting between VxWorks filesystems and low level block device drivers, however it was only a "glue and pass-through" layer in that there was no bio scheduling performed. This was one significant limitation for VxWorks filesystems found during working with SATA and USB 3.0 UAS/UASP drivers, where TCQ/NCQ/USB 3.0 Bulk Streams and scatter-gather capabilities of the underlying hardware can be utilized to enhance the performance. To address this and other limitations for VxWorks filesystems, I took the responsibility to design and implement a Pluggable I/O scheduler Framework, which had the following features: 1) Pluggable and modular design that can be included/excluded as needed; 2) Core I/O Scheduler Framework supporting bio reordering and merging; 3) Support several major schedule policies (such as NOOP, Deadline, SSD) that can satisfy requirements for different storage media (e.g, Magnetic Disk and SSD); 4) The design was a good exercise of delegation design pattern. The Pluggable I/O Scheduler Framework has been delivered as a fully supported Wind River product. The performance was tested to have greatly improved than previous VxWorks versions that had no such enhancements.

- **Topology Aware Power Management Framework:** Power Management is a feature that is disregarded by most people but in fact it should be respected as a challengeable and valuable feature in almost all modern computer systems. Earlier versions of VxWorks offered some basic Power Management features, but it does not address most recent advancements in this field. I took the responsibility to redesign and implement the VxWorks Power Management Framework in a systematic approach. It supports the following features: 1) Runtime device power management subsystem, which puts idle devices into low power state automatically by an idle timeout mechanism; 2) Platform power resource management subsystems, including clock domain, voltage domain and power domain subsystems; 3) CPU DVFS domain and IDLE domain management subsystems; 4) CPU utilization measurement subsystem; 5) CPU tickless idle management; 6) System power management subsystem; 7) CPU topology reporting framework, supporting CPU topology hierarchy levels from NUMA node, CPU package, CPU cluster, CPU core, CPU thread; 8) Power Aware SMP CPU task scheduler, supporting power aware load balancing (considering both DVFS domain and IDLE domain features) and recent industry trend for performance asymmetric architectures (HMP) such as ARM big.LITTLE. This work has been delivered to an EAR customer based on i.MX6 platform, and it is actively processed under formal production procedures.

- **Internet of Things Research on Security**: Recently I am doing research on Internet of Things. The current focus is on Operating System Security, with an intention to bring in a flexible security framework (implementing access controls such as DAC, MAC, as well as RBAC), which would further secure VxWorks in all its subsystems, so that VxWorks enabled systems can be guaranteed to be safe and secure in the IoT world.

- **2009.08-2010.12, Senior Engineer, China Development Center of Wind River Systems**

Due to my outstanding contribution to the USB product line, I was promoted to be Senior Engineer in October 2009, 1.5 years after working with Wind River Systems. I became the technical lead in USB team since then. Beyond routinely providing support for worldwide customers (fixing defects and adding small feature enchantments), I was responsible for designing and implementing big new features that satisfy customer requirements. The following are some high lights of the outcome during this period.

- **Wind River USB Stack LP64 Support**: One major new feature that was added during this career period of mine was the transition of the Wind River USB stack to be LP64 compatible in support of 64 bit VxWorks transition. Although it seems to be simply a change from 32 bit to 64 bit support (involving 64 bit code adaption), it actually included several other big changes to the whole Wind River USB stack (mainly due to change of memory model in the Core OS). One of the key changes was to update the whole stack to use vxbDmaBufLib (a general DMA buffer handling library for VxWorks) to handle all kinds of DMA related buffers, by doing this, we unified DMA buffer handling in the low level drivers, such as DMA bounce buffering, cache operations, alignments, etc. This work required very detailed understanding of the existing Wind River USB stack, especially the HCDs (EHCI, UHCI, OHCI) in order to modify them to be compatible for both 32 bit and 64 bit systems. In doing the work, I also participated in discussions with NA team and made key idea contribution for improving the vxbDmaBufLib itself.

- **Fully Featured USB On-The-Go Stack**: As a supplement to the USB 2.0 specification, USB OTG (On-The-Go) has becoming a widely supported hardware feature in recent embedded platforms due to its flexibility to support both USB host and device roles on a single USB port with dynamical role switch capability. Another major new feature that I created was a totally new USB OTG stack. The new USB OTG stack was a modular and component based design, layered with user friendly API interfaces, supporting all USB OTG features (SRP, HNP, HNP Polling,…). More importantly, although the new stack added about 10K lines of code (with about 3K lines of comments), it only required minimum changes to existing USB stack, and it was backward compatible with existing USB usage models and support new OTG based usage models. The USB OTG Framework core was a good exercise of state machine design pattern, handling tens of event changes and state transitions. This work has been delivered as a formally supported Wind River product.

- **Translating USB 3.0 Specification to Chinese**: As an intention to fully understand the newly released USB 3.0 specification, as well as trying to spread the USB knowledge to engineers working in this field, I spent my spare time during almost a half year to translate the 500+ pages of USB 3.0 and UASP specifications into Chinese. The translated USB 3.0 specification has now been posted online here http://www.cnblogs.com/coryxie/category/596771.html.

- **2008.03-2009.07, Engineer, China Development Center of Wind River Systems**

During the first 1.5 years working as a development engineer in the USB team at Wind River Systems, I was mainly responsible for maintaining and improving the existing Wind River USB stack. These efforts received very good feedbacks from worldwide customers, thus the USB team was awarded the “Business Focused and Customer Driven Award” by the company. With these experiences, I gained expertise in almost every aspects of USB, including HCDs (EHCI, UHCI, OHCI), USBD and Class Drivers. I soon became the “go to person” for almost all USB related questions in the company. The following are some high lights I have done for VxWorks.

- **Cumulative Patches for Wind River USB Stack**: By handling TSRs (Technical Support Requests) and Escalations from worldwide customers, I gained in-depth knowledge and experiences for various parts of the USB stack as well as the VxWorks RTOS itself. Based on historical defect fixes and enhancement changes, we created series of patches called USB CUM patches for various released versions of VxWorks from VxWorks 5.5.1 up to VxWorks 6.6. With these patches, we soon cut down the level of support requests from customers.

- **GEN2 Class Drivers for Wind River USB Stack**: With the insight into the existing USB stack, we developed a new generation of USB class drivers called USB GEN2 class drivers, including USB Mass storage class, USB HID class (Keyboard and Mouse), USB CDC class (Networking), USB Printer class. I was responsible for the USB GEN2 Core Framework, as well as designing and implementing of USB GEN2 Mass Storage Class Driver. The GEN2 MSC class driver was a layered and modular design (e.g, it has an independent SCSI command handling module), by way of innovative ideas such as “free bandwidth media hot plug check”, it addressed several key limitations of the GEN1 driver. With these new GEN2 class drivers, we deprecated the legacy GEN1 class drivers in latter VxWorks releases.

- **2007.8-2008.02, Engineer, Platform Department, Lite-On Technology Co., Ltd**

Although I only served Lite-On about half a year, but the Linux BSP development was a much appreciated job for me.

- **Linux BSP Development**: Acting as the company's Linux technical lead, I was responsible for the company's ARM926EJ-S based SOC platform Linux BSP development, including the use of MontaVista Linux RT Preemption Patch to improve the original kernel and drivers, USB Host and Device Driver as well as u-boot porting, etc. Moreover, I was also responsible for BSP development for Sigmatel DC2250 based board, through which I also gained some knowledge and experiences of the ARC architecture (a variant of SPARC).

- **2005.5-2007.6, Engineer, Platform Department, Beijing Destiny Electronic Technology Co., Ltd**

During the 1st service period at Destiny (and the 1st period of my career), I was working as a BSP engineer for VxWorks BSP and peripheral device driver development. I am quite thankful to my 1st manager who gave me a lot of chances and guidelines to do low level coding and debugging, which gave me solid foundation to work as an Embedded Systems developer. Especially, by doing these trivial things, I gained in-depth knowledge and concrete practices of mainstream embedded operating systems such as VxWorks, ThreadX, and Linux.

- **VxWorks BSP and Device Drivers**: The key responsibility for me was developing VxWorks BSP and device drivers for printer control boards. In doing the work, I actually used the low level hardware features (e.g, MMU and Cache) of PowerPC, MIPS, and ARM embedded processors. I also gained hands on experiences using assembly languages of various architectures to initialize memory system and hardware modules, as well as device driver development with Hard Real Time Operating Systems. I also learned many commonly used components/interfaces and building blocks of embedded systems, including the USB Host/Device, Ethernet MAC/PHY, DDR2, FLASH, NVRAM, etc.. I also gained capabilities on the use of debugging and measurement tools (such as ICE and oscilloscopes).

- **PictBridge Direct Printing System**: As one of the new functions for digital cameras, PictBridge enables users to connect a DC directly into a printer and perform image printing. The key enabler for this functionality is the USB Still Image Capture Device Class as well as the Picture Transfer Protocol. I was responsible to create a USB driver stack (including support of OHCI and USB SIC class) to enable this functionality. The driver stack has been successfully delivered in a commercial product.

During the 2nd service period with Destiny, I was working on ZORAN ZR4230 (ARM1136JF-S based) platform, taking on the following responsibilities:

- **GDB Agent for ThreadX RTOS**: Threadx is an RTOS for embedded systems, but at the early times when we used it for a printer controller board, it had no flexible way of debugging other than using hardware based ICE. The support of network based debugging mechanism was deemed one of most desirable features from App designers in the company. I offered to take the responsibility upon myself to create a GDB based debug mechanism for the ThreadX RTOS. I designed a GDB Agent (as a high priority thread of ThreadX) following the GDB Remote Serial Debug Protocol and using an undefined ARM instruction to create software breakpoints. This enabled the App developers to show ARM registers, memory variables as well as examine memory addresses, from a standard GDB remote connection on debugging host.

- **Bootloader for ThreadX RTOS**: I was also responsible to port u-boot to the board for easier download/boot/debug of ThreadX. I mainly completed the following work: 1) Architecturally ported u-boot to this platform (at that time of the work u-boot had no ARM11 support yet); 2) Modified u-boot to support ThreadX operating system downloading and executing; 3) Modified ThreadX OS scheduler to save and restore ARM VFP registers thus fixed a severe bug of this RTOS (as an extra effort).

- **Linux Kernel Porting for ARM1136**: Other work on this platform also included architecture/chip level porting of the Linux 2.6 Kernel and setup a basic application environment using busybox. This works was completed by having a basic kernel with serial console working as a Proof of Concept delivery, without other device drivers developed (because I went to Lite-On later).

OTHER PROJECTS

- **2004.7-2005.8, Student, Embedded Systems Lab. of Beijing Information Technology Institute**

As a member of a 3 student team, representing our college, I participated in the National Undergraduate Electronic Design Contest-Embedded System Design Invitational Contest during the summer of 2004. My college graduation project was also based on the same system used during the contest.

- **Embedded System Design Invitational Contest**: We designed a "Linux-based Networking Video Conference System" using the Intel XScale PXA255 Sitsang development platform. The project was based on Intel IPP Integrated Performance Library (G.723 and G.711 voice compression), implementing the H.323 standard. I was acting as architect role during the project, responsible for the overall system design, as well as the Linux kernel device drivers for a USB host controller (ISP1161) and a USB camera (Web Eye 2000, OV511 Image Sensor); Latter in the development phase, I was also responsible for the GUI (Qt based) and Embedded database (SQLite based) development. Although it has something to be improved, our system was selected as a winning project for the 3rd prize.

- **College Graduation Project**: During my graduation project, I conducted the implementation of "Linux-based Networking Video Conference System Optimization under Wireless LAN environment" based on the aforementioned Intel PXA255 Sitsang development platform. The project mainly completed a Symbol Spectrum24 CF Wireless LAN adapter driver, and the network application part has been adjusted and optimized so that the system can meet wireless LAN environment.

- **2005.9-2005.10, Open Source Contributor, Domestic Embedded Open Source Project LUMIT**

To take advantage of spare time, I participated in a domestic embedded open source project called LUMIT (Let Us Make It Together).

- **Let Us Make It Together**: The project was based on a Samsung S3C4510B board. I ported u-boot, the widely used bootloader in embedded Linux systems, to this board. On that basis, I ported a Linux kernel 2.6 based uClinux to the board, including the latest version of the MTD, added JFFS2 filesystem support for the FLASH part. Through participation in the project, I met a number of embedded system developers and we became friends. Talking about issues we met, sharing of success we achieved, we were fully aware of the benefits of open source and open thoughts.

- **2006.3-2007.09, Technical Consultant, Artificial Intelligence Lab. of Tsinghua University**

A friend of mine invited me to act as a technical consultant to participate in a National High-Tech Research and Development Program of China (863 Program) project, to assist Ph.D candidates in their conduction of ZigBee based researches.

- **Fully Featured Linux based ZigBee Protocol Stack**: Initially I worked for S3C2410 based Linux ZigBee base station; Later, I assisted these Ph.D candidates to port TinyOS to S3C2410 and i.MX31 platforms. The i.MX31 platform was eventually used as the ZigBee Coordinator, with which I assisted these Ph.D candidates for the Linux BSP development. Based on that Linux platform, I designed a Linux based ZigBee Protocol Stack (SPI connected CC2420 2.4 GHz IEEE 802.15.4 compliant RF Transceiver). The stack implemented ZigBee MAC/NWK/APS layers, following IEEE 802.15.4/ZigBee specifications. By working on this role, I became familiar with ZigBee Wireless Sensor Networks. The system was eventually used in certain 863 robot security system, and successfully passed acceptance.

- **2010.11-2011.01, Open Source Contributor, CellOS, A Multiboot Compliant X64 SMP Operating System**

With the popularity of 64 bit Multi-Core Intel systems, I enthusiastically felt there is a need for me to learn the new trends in Intel architecture. I then created an Open Source project for a hobby Operating System named CellOS.

- **Homebrew Multiboot Compliant X64 SMP Operating System**: The project was initially hosted on https://xp-dev.com/svn/cellos/, but has been moved to https://github.com/CoryXie/CellOS in 2013. The CellOS project was created with the following features: 1) Multiboot Compliant, boot on GRUB legacy; 2) Works in Intel 64 IA32e (long mode); 3) Born to support SMP; 4) Basic memory management (basic page allocator with a port of TLSF heap allocator); 5) Schedule class based scheduler framework (with RR and FIFO policy implemented); 6) In kernel POSIX API support (actually I created a lot of headers based on POSIX specification from http://www.opengroup.org/); 7) Basic kernel objects such as Spinlock, Semaphore, Mutex, etc; 8) Integrates Intel ACPICA library for device management; 9) Support for basic X86 hardware, including APIC, PIT, HPET, PMC, etc; 10) High Resolution Timer Management Framework; 11) Supports a kernel shell with commands implemented in a u-boot like fashion; 12) Runs on real hardware, as well as QEMU and VMware Workstation (including support for VMware Pseudo Performance Counters). The project, although demonstrated as a playground for Operating System design and research, has been suspended in its development due to my new ambitions.

- **2013.05-2013.07, Open Source Contributor, Open Source Projects Analyzing and Chinese Commenting, Multimedia Lab. of Tsinghua University**

  - **Open Source Projects Analyzing and Chinese Commenting**: By an invitation from a professor from Tsinghua University, I participated in another National High-Tech Research and Development Program of China (863 Program) project, which was for the analyzing and commenting (with Chinese) of tens of key enabling open source projects (http://124.16.141.148/index.php). I was responsible for the following projects: 1) Analyzing GRUB legacy (0.97), with Chinese Comments (https://github.com/CoryXie/GrubLegacy) and Analysis Reports; 2) Analyzing GRUB 2, with Chinese Comments (https://github.com/CoryXie/GRUB2) and Analysis Reports; 3) Analyzing Sysvinit (sysvinit-2.88) with Analysis Reports; 4) Analyzing Libvirt (libvirt-1.0.0) with Analysis Reports. These projects gave me much deeper understanding of how modern Operating Systems work. The reports are not currently publicly available yet but can be provided for interested entities.

- **2013.07-2014.01, Open Source Contributor, Android Native Windowing System Based on Skia, Multimedia Lab. of Tsinghua University**

  - **Android Native Windowing System**: Android has been most popular Open Source Mobile Operating System which featured a Java based App development environment, enabling tons of Apps available on the market. However, Java is not always the best option in all cases (for performance reasons, or for license reasons). One (not to be named) domestic vendor wants to use the basic Android framework created by Google, but would like to get rid of (or use it side by side) the Java environment. So it requires a native windowing system based on Android frameworks. I was invited by the same professor from Tsinghua University to help them identify the feasibilities of making a native windowing system (not the Android NDK development environment which is still initiated from Java). With that intention in mind, I created SkiWin project (https://github.com/CoryXie/SkiWin) which is an Android native windowing system based on Skia (2D Graphics Library used by Android in its native layer for rendering graphics). This includes features like: 1) True native App starts from main in C language; 2) Basic View creation and management; 3) Supports input event handing (native InputManager) with the views; 4) Based on Skia which is from Android Framework, thus it can live with Android Java environment, switched by a switcher Android App (https://github.com/CoryXie/SkiWinSwitcher).

KEY CAPABILITIES

The following list shows the key technical aspects that either I am quite expertized on or have enough know-hows.

- Mainstream operating system kernels: Linux, FreeBSD, VxWorks, ThreadX,…
- Mainstream CPU architectures: Intel x86, Intel 64, ARM, PowerPC, MIPS,…
- Common CPU architecture knowledge: Cache Coherency (MESI, MESIF), Memory Consistence (Memory Ordering, Write Buffers), MMU (Segmentation, Paging, TLB), Virtualization (VT-x, VT-d),…
- Common protocols and standards: USB 2.0/3.0 (xHCI, EHCI, OHCI, UHCI,…), PCI/PCIe (RC, Bridge, Switch,…), SATA (AHCI), SCSI (USB MSC, iSCSI), IPv4/IPv6 (Ethernet, Linux Networking Stack), Bluetooth, ZigBee,…
- Mainstream development languages: ASM (ARM, X86, PowerPC, MIPS), C, C++, Java, Python,…
- Common devices and interfaces: Ethernet, I2C, SPI, UART, NOR/NAND (MTD, FTL,…), SSD, DDR,…
- Common hardware debugging tools: Oscilloscopes, Logic analyzers, USB analyzers, ICE tools,…
- Frequently used development tools: GNU Linux (bash, make, git, autotools,…), Eclipse, IDA Pro,…
- Mainstream development processes: Waterfall, Prototyping, Agile,…

SELF ACCESSMENT

I am truly a technical person, with a very strong intention to gain “full stack expertise” in the field of System Software. I have strong interests in low level technologies (as in how Cache Coherency works) that enable high level technologies (as in how Internet-of-Things works). With these low level knowledge bases as well as enough engineering experiences, I believe I could achieve my career goals. However, technology is not everything that defines me. In fact, I would expect the following personal traits should finally define me and my career.

- Positive, sincere, well-motivated, and open-minded.
- Can endure hardship, with a strong sense of responsibility.
- Have a strong ability for self-learning and independent analysis.
- Good at system-level thinking, with a strong spirit in team working.