Andrew L. Hanusch

Senior Software/Firmware Development Engineer
619-793-9530 ● andrew@hanusch.email ● LinkedIn

Objective

For more than 30 years I have provided quality professional software and firmware engineering, development and management services to many well-known California companies. I have successfully managed, built and deployed several large-scale distributed applications, mobile applications and embedded firmware solutions that consistently have had a positive impact on the financial success of my clients and employers. I am currently accepting direct or contract software engineering positions that will effectively leverage my knowledge of computer science, my advanced problem-solving abilities, and my extensive industry experience. I am highly motivated and am prepared to make an immediate and positive contribution to any situation.

Skill Summary

Desktop/Server Operating Systems	Linux (all types), Android, Windows (all types), OSX/MacOS
Embedded Operating Systems	Linux, Android, ThreadX, FreeRTOS, VxWorks and bare metal (no os)
Programming Languages	C/C++, Kotlin, Java, .NET (C# and VB), JavaScript, PHP and Python
Assembly Languages	IA32, x64, Hexagon DSP (QDSP6), ARM and SMALI (Java Byte Code)
Web Application Technologies	LAMP (Linux, Apache/NGINX, MySQL, PHP), JavaScript and HTML5
Cloud Platforms	Amazon Web Services (AWS), Google and Azure Cloud Services
LinkedIn Skill Assessment Badges	C, C++, Python and Linux

Professional Experience

Flock Group Inc. dba Flock Safety

From **May 2024** to **May 2025** as a Senior Development Engineer I lead the connectivity (LTE/NR, Wi-Fi and Bluetooth) engineering team's effort to build the next generation license plate reader for Flock Safety with Linux/Android as the target OS. I was responsible for all (Linux/Android) connectivity device drivers, Android system daemons, startup scripting and SE Linux rules.

I was responsible for adapting the reference Android Radio Interface Layer Daemon (rild) for LTE/NR network connectivity control using Quectel modem modules. This effort included determining correct AT modem commands for specific Quectel modems and message handling from the Android Telephony layer to provide signal strength, statistical and configuration information to the company's "Phone Home" Android application.

I also created and maintained the Subscriber Identity Module Daemon (simd). The simd monitors the insertion status of the SIM card in the license plate reader device. The simd is responsible for switching between the external physical SIM card and the internal eSIM module. The simd monitors the insertion status based on an GPIO interrupt managed by the "sim-detect" device driver. The simd also allows selection from among the various installed provider profiles on the eSIM module.

Additionally I maintained the modem parts of the "Phone Home" Android application to reflect changes to the rild. This allowed installers to use the information as a site survey when determining device placement. The "Phone Home" application's information is also sent to the Flock Cloud back-end for viewing on the Flock Web Portal.

Radio Interface Layer Daemon and Subscriber Identity Module Daemon development was done using C and C++. Phone Home application development was done using Kotlin.

Deer Management Systems, LLC dba Tactacam and Reveal Cellular Cameras

From **September 2022** to **May 2024** I led the embedded software engineering team at Deer Management Systems (DMS). As a Senior Research and Development Engineer I led the company's effort to build an internal engineering team in order to reduce dependency on external vendors and maintain control of intellectual property.

I led the firmware development for DMS's first security camera. The camera is based on the Ingenic T31 (MIPS CPU) SoC and GC2064 image sensor. Main camera firmware was hosted on embedded Linux and implemented in

C, C++ and bash scripting. The platform also contains an ARM MCU that controls GPIO interrupts from external devices and external Bluetooth communications. The MCU firmware was implemented as a bare metal application. The camera operation is activated by a PIR (passive infrared) motion sensor or button press. Upon activation the camera captures images (JPEG) and/or video (MP4 encoded h.265), then uploads the results to AWS using a Quectel Cellular module or a SiliconLabs WiFi radio. I led the team in image capture, AAC and MP4 encoding and AWS functionality (S3 and MQTT communications). I was also solely responsible for the firmware optimization effort. I improved the firmware/camera performance from capturing (and uploading to AWS) 1850 images per battery charge to more than 5000 images. I also designed and created the end of line manufacturing tests for product delivery.

I also worked with multiple external vendors to create the company's first bird feeder camera using the Ingenic T31 SoC and the next generation of trail and security cameras using the Rockchip RV1106 SoC. I participated in the initial hardware design efforts evaluating and recommending potential parts. I also provided reference firmware in an effort to train the external engineers in multiple areas (image capture, MP4 encoding, communications (LTE, Wi-Fi, Bluetooth) and AWS. I also led the internal hardware and firmware validation efforts.

I was also responsible for testing and validating new firmware releases provided by external vendors for the company's existing (1st and 2nd generation) trail cameras.

Google LLC [McAfee/Trellix] (Triple Crown Consulting, LLC)
From **October 2021** to **July 2022** I was a member of Google's 'Android SDK Research' team (part of the Google Play Protect project). At Google I was tasked with performing security analyses on Android SDKs. This effort consisted of creating test/sample applications based on the SDK under review, de-compiling the resulting APK using tools such as apktool and jadx (and Google's internal proprietary tools), and then validating any security issues discovered using Google's static and dynamic analysis tools. The test applications were required to exercise as much of the target SDK as possible. I was responsible for all aspects of application development. I performed all application design, implementation, and testing. A complete development lifecycle was required to ensure that a true analysis of the SDK could be performed. The SDKs reviewed included advertising, analytics, remote file access, a/b testing, language translation, multiple Google Firebase SDKs and more. The test applications were created using Android Studio and Java.

I also created a SMALI code instrumentation system of python scripts (targeting Linux and MacOS host platforms). This system de-compiled the target APK, extracted the SMALI source code for the SDK, created a new SMALI project, added debugging instrumentation as desired, compiled the project into a JAR artifact and then copied the JAR file to the correct file system location for the original APK's linking step. This instrumentation allowed dynamic analysis of code coverage, call-stacks and various data dumps.

The Google Android SDK Research team project was completely remote. All coordination was accomplished using Google tools.

Phase 3 Microsystems, LLC. (Independent Contractor)
From **May 2017** to **September 2021, May 2014** to **December 2014** and **January 2003** to **May 2006** I provided IT and software engineering consulting services to several small and medium sized West Coast companies. I was responsible for the software development needs of our IT service clients. This included diverse projects such as Android app development (Java), Android app reverse engineering and modification (smali), embedded real time device drivers and controllers, OTA update daemons for embedded processors, C/C++/C# Windows desktop applications, custom Android builds and various big data applications (C and Python). I have also managed several data center build and remodel projects where I was responsible for all cabling, server conditioning and hardware integration.

I also contributed to and managed several full stack LAMP projects. I provided both back-end (server side PHP scripts) and front-end (browser based JavaScript apps) solutions for multiple web applications.

H4 Engineering, Inc. dba SoloShot (VIA Technical, LLC)
From **January 2017** to **September 2017** I was responsible for all low-level Linux embedded systems. I created a custom secondary boot loader (SBL1) for the Qualcomm Snapdragon 410 SoC (msm8016) to initialize the SoloShot base, load various (non-Linux specific) boot images and continue the boot process. The SBL1 was built on Ubuntu Linux and cross-compiled for ARM using the ARM RVCT compiler suite. I implemented an ExFat kernel file system driver for Linux Android and heavily modified the vold (Volume Daemon) to take advantage of multiple SD Card file systems during auto-mounting. I also implemented an I2C Linux device driver for the Toshiba TC358743XBG HDMI

to CSI (MIPI Camera Serial Interface) bridge chip. This included modifying the msm kernel audio subsystem to allow the correct I2S audio format. I also modified the Little Kernel (apps bootloader) to initialize the QuickLogic BX5BxA DSI (MIPI Display Serial Interface) to RGB bridge chip. I also created several Android Apps (Java and C++) for testing of the above kernel device drivers.

Verifone Systems, Inc. (NESC Staffing, Inc.)
From **June 2016** to **December 2016** I provided bug fixes and functional enhancements for the company's payment agent software running on a variety of Verifone payment terminals. All target development was done for an embedded security enhanced Linux based on the buildroot project. Development was done in C/C++ and cross-compiled using GNU tools. I also provided HTML5 user interface bug fixes and enhancements (all Verifone payment terminals use HTML5 for user interfaces).

Panasonic Avionics Corporation (MLS Technologies, Inc.)
From **September 2015** to **June 2016** I contributed to the company's In-flight Entertainment system by creating the Service Discovery and Event Server middle-ware applications. Together these applications allow a seat-back or other passenger device to query for currently running services and then to be notified when any services change. These applications were developed in C/C++ for Linux using a REST API and FastCGI through an NGINX server.

I also created a series of prototype/proof-of-concept apps for Android using JNI to provide fast access to binary files containing media information. I created the Android user interface (Java), the JNI shared library (C/C++) and the binary file format that allowed C structures to be serialized and deserialized directly without any additional parsing.

GoPro, Inc. (Vertisystem, Inc.)
From **December 2014** to **July 2015** I positively contributed to multiple projects. I provided several bug fixes and user interface enhancements to GoPro's Android and iOS camera control apps. I was also responsible for the creation of a test framework used to validate the external (WiFi and Bluetooth) interfaces to several models of GoPro cameras (including four new cameras currently in development). This included all aspects of camera control as well as validating JPEG and MP4 creation and validating GoPro's MPEG-2 live streaming protocol (live streaming from camera to the GoPro mobile app). The test framework was compatible with Windows 7+, Mac OSX, and Linux. All software was written in C/C++ and included the integration of the Python 3.x interpreter for easy test creation/scripting.

Qualcomm, Inc. (RJT Compuquest, Inc. and Technology Locator, Inc.)
From **February 2010** to **May 2014** I created the Target Image Simulation Environment (TISE) that allows direct testing of the audio, voice, video and sensor functions of the Hexagon DSP (QDSP6) in a Windows 7 desktop environment. My software (running in the HLOS) provides the communication infrastructure (via shared memory) required for DSP control. It also models several other CPU cores, external peripherals, ASIC functions and memory buses that are present on the Qualcomm Snapdragon product line. My software is used internally by Qualcomm developers and is also released externally to OEM and ISV customers (for use in development and testing of new products). It is also distributed as part of the Hexagon SDK program.

I also provided direct technical support to OEM and ISV customers and bug fixes for the Qualcomm board support package, audio, voice, video, and sensors teams. This support included all levels of JTAG and Trace32 training and scripting as well as generic embedded software optimization in C and C++ for all customers on the Hexagon DSP and Snapdragon SoC.

Also as part of the TISE project, I created a diagnostic tool that communicates directly with an Android or Windows phone. This tool collects diagnostic and performance data from all CPU cores on a Snapdragon SoC via a USB connection (Android) or via a TCP/IP connection (Windows and TISE). The diagnostic and performance data is then displayed for human monitoring and logged for offline processing / analysis.

I also created a test framework for Qualcomm's Hexagon DSP (QDSP6). The software consisted of command line applications designed to run on the Linux Android platform. These applications exercised all audio and voice functionalities of the Hexagon DSP in an HLOS agnostic way. I also created the Linux device drivers used to control the audio and voice features of the Hexagon DSP (QDSP6) and an Android UI based test launcher app. The Android app first discovered the capabilities of the device and then presented the user with a list of applicable tests. The app then collected all test output and logged it for off-line processing. I maintained this app for the 8650 and 8660 versions of the Snapdragon SoC.

From **October 2008** to **May 2009** I was responsible for the board level testing software for the company's ASIC testing equipment (radio interface simulation board). My software communicated directly with Qualcomm's custom hardware (for control) and with the test measurement equipment (for data acquisition). Device control and data acquisition software was implemented using Visual C# (.NET) and C++.

From **May 2008** to **May 2009** I designed and implemented software to convert raw test output into human readable Excel Spreadsheets. This included determining which units failed and which passed both the calibration and validation testing phases and presenting performance information in a graph format. The spreadsheets were created through COM Automation (IDispatch interface) with C++. User interfaces were created using C# (.NET).

From **November 2007** to **May 2008** I created control software for the company's HSPA+ ASIC testing equipment. The software communicated to test equipment via USB or Ethernet. It included a graphical user interface written in C++. I also created the firmware for the FPGA controller board using Verilog.

From **December 1997** to **June 2000** I contributed to a miniature CDMA infrastructure project. I was responsible for all network communications software. I created a distributed console I/O library for embedded applications using TELNET. I built an NTP server (and client) that interfaced to a Brandywine GPS receiver (I created the embedded VxWorks kernel driver for the Intel i960) providing real time of day information and synchronization (within 2 ms) on the network. I developed a SIP Server for use in a VOIP application. I also developed a "Mobile Switch Controller" to circuit switch voice traffic on a network. All software was required to run on Linux, Windows NT, Solaris and VxWorks (embedded Intel i960 and TI DSPs). This project was funded by the Department of Defense.

From **December 1997** to **June 1998** I participated on a design and code review committee for a distributed client/server cellular phone diagnostics utility suite. I was the lead Windows NT architect for the project's design phase. This project eventually became the Qualcomm Product Support Tool (QPST).

From **June 1997** to **December 1997** I worked on a cellular network-planning tool. I developed the GSM analysis module for the existing Solaris product as well as a PostScript printer driver for the Solaris operating system. I also co-wrote a Software Development Manual which outlined C++ coding standards and guidelines.

University of California at San Diego
From **June 2001** to **December 2001** I taught the Embedded Networking class in the UCSD Computer Engineering department. The class explored all layers of the OSI model. Protocols covered included Ethernet, Bluetooth, ATM, CDMA, GSM, HDLC, OC3, IP, TCP, UDP, SNMP and X10 (home appliance control). I also introduced basic implementation issues for various embedded platforms. The course was part of the Embedded Certificate Program.

Education

Udacity Nanodegree in Deep Learning
Introduction to Deep Learning (Fall 2024)
Convolutional Neural Networks (Spring 2025)

Georgia Institute of Technology (Georgia Tech)
Various graduate courses in Computer Science [Machine Learning and Artificial Intelligence] (2020 to 2024)

University of Washington and National University
Bachelor of Science degree majoring in Computer Science (August 1999)
4.0 GPA in courses required for major. 3.96 GPA overall