Applications of Waveform Virtualization on High-Performance Computing (HPC) Platforms using High-Level Coding Languages

Michael Beeler, Kasra Toyserkani, & Michael Geist, *Advanced Technology Group (ATG), Envistacom*

***Abstract*—This paper studies the application of satellite, tactical radio, or terrestrial radio waveform virtualization on commercial-off-the-shelf (COTS) High-Performance Computing (HPC) platforms using a high-level software programming language and an open computing framework to provide hardware abstraction. The method of virtualization described support the integration of data analytics and interference mitigation techniques, while delivering performance comparable to "purpose built" wireless communications devices. Virtualizing a modem waveform on a COTS platform was not possible prior to the introduction of the hardware assisted Central Processing Unit (CPU). The HPC server utilizes the combination of a CPU with an integrated hardware acceleration device, e.g. Field Programmable Gate Array (FPGA). The introduction of the open and heterogenous computing framework known as Open Computing Language (OpenCL) provides the mechanism to utilize high-level C with OpenCL extensions as the programming language with the ability to provide advanced task scheduling of tasks based on the need for CPU (general data processing) versus FPGA (compute-intensive) processing. The combination of the HPC with OpenCL has demonstrated that high-performance waveforms can be virtualized and hosted on COTS HPC servers. Examples of such waveforms that can be supported in this new architecture are Digital Video Broadcast – Satellite version 2 (DVB-S2) [1] and DVB-S2x [2]. The OpenCL framework provides a hardware abstraction layer to remove dependence on the underlying hardware. Furthermore, the combined solution delivers performance comparable to purpose-built hardware. Finally, there is no longer a need to be locked to a given hardware technology or hardware vendor, and the underlying hardware version/generation and type/family can be upgraded with minimal to no impact to the end user.**

Keywords—HPC, OpenCL, Virtualized Waveform, and High-Level Coding Language, Cloud Computing, Data Analytics, Public and Private Data Centers

I. Introduction – Hardware and Software Research

The concept of running a waveform on a COTS platform started as a result of a US Army Phase I Small Business Innovation Research (SBIR) program A16-113 that called for an innovative way to support a modem on a standard Personal Computer (PC) hardware architecture. The primary objective of the research was to determine if supporting a waveform with a COTS PC was possible. While CPU processing has increased over the years, the standard CPU is still not capable of supporting multi-hundreds mega-bit per second (Mbps) performance on a non-hardware assisted PC. The study explored both hardware solutions as well as compiler strategies and the HPC with OpenCL was identified as a possible solution. In the Phase I SBIR, it was demonstrated that a DVB-S2 [1] waveform's most compute intensive functions could be supported and yielded +100 Mbps of performance. A Phase II SBIR was awarded to carry the research from the compute intensive cores to a fully functioning DVB-S2 modulator and demodulator. The Phase II SBIR award was combined with independent research and development (IR&D) from Envistacom to fulfill this objective.

Historically, the evolution of the modem technology has been exclusively focused on purpose-built solutions with a Printed Circuit Board (PCB), CPU, FPGAs, discrete components, and/or an Application Specific Integrated Circuit (ASIC); and combined with implementation code targeted for the custom hardware environment to deliver a limited set of capabilities and a single or limited set of waveforms.

Building on work done on the Digital IF standard effort performed by the US Army's CERDEC Space & Terrestrial Communications Directorate (STCD) CERDEC FAST working group, Open Standard Digital-IF Interface (OSDI) for SATCOM Systems [3] was utilized as the transport of the post modulated waveform. This development focused on fostering open market competition while not limiting creative "black box" designs, non-proprietary technology, and designs that achieved compatibility-interoperability-interchangeability. This effort resulted in a standard known as the TIA 5041 OSDI and provided a modular architectural framework defining the signal processing elements and the subsystem communication interfaces to create a class of digital intermediate frequency (IF), or "all digital," strategic fixed SATCOM terminals.

The OSDI Digital IF interface was used as a means of transporting digitized I/Q between the HPC hosting a virtualized waveform application and a TIA 5041 compliant Digital Conversion Subsystem (DCS) to generate the RF signal. This DCS, also referred to as an "Edge Device", would be a waveform agnostic and multi-band RF conversion device that could reside locally with the HPC or remotely at a teleport over a network.

II. Introduction of the HPC

The HPC was introduced as an alternative to "throwing more iron," e.g. more CPUs at the processing limited server farms with the desire to improve the processing per Watt ratio. The next logical alternative to improving the processing power was through the introduction of hardware assist. The term "HPC" was added to use the best a CPU had to offer but add a hardware acceleration engine (e.g. Graphic Processing Unit

(GPU), Digital Signal Processing (DSP), or FPGA) to provide acceleration for computer intensive algorithms and mathematical operations. The FPGA is the latest form of hardware acceleration technology that has proven to offer the highest performance/Watt. Thus, it is being rapidly deployed in Public Data centers to add more computing power for applications running in the Cloud.

While the HPC was originally designed for batch type processing where large volumes of data were passed to the HPC for offline processing, the HPC needed a mechanism for supporting a real-time and continuous flow of data such as with a communication signal. The original research found that while the HPC hardware was suited for supporting real time flow of data, the compilers and framework were not quite ready. The early research found that OpenCL could be "steered or guided" to make the distinction of real time versus non-real time processing. It was not until the concept of Host Channels were introduced that perpetual threads could be brought to bear for the support of a real time modem flow to support user data, forward error correction (FEC), data to symbol mapping, and modulation

Two manufacturers have become the dominant providers of FPGA-based hardware acceleration to support the OpenCL architecture: Intel (Altera) and Xilinx. Both have become the industry standard being installed in nearly all the HPC servers being shipped throughout the world. The architecture of the HPC with FPGA-hardware acceleration is shown in Figure 1.

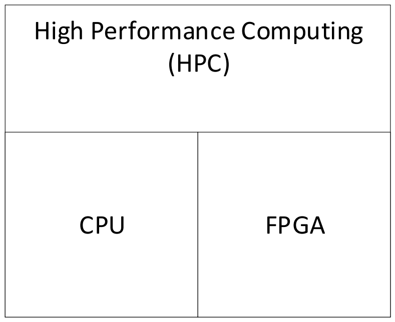

Fig. 1. HPC Hardware Architecture

III. Introduction of OpenCL

While the hardware is instrumental to the virtualization effort, another piece of the architecture that is tantamount to the hardware is the coding environment. The OpenCL coding environment allows one to utilize high-level software with directives and extensions to move sections of the code that are better suited to be supported by the hardware assisted processing while saving less compute-intensive processing for the CPU. While OpenCL offers a high-level coding environment, the compiler technology is such that one still must be savvy on hardware to fully appreciate and utilize the hardware assisted processing.

The OpenCL is the framework for developing applications that are hardware agnostic (platform or hardware independent). In fact, OpenCL was defined to be able to execute across heterogeneous-computing platforms that consist of CPUs, GPUs, DSPs and FPGA based acceleration cards. OpenCL includes an Application Programming Interface (API) for controlling the platform and abstracting the underlying hardware. The OpenCL specification is currently at version 2.2.

The OpenCL framework can be utilized to implement the digital processing of any telecommunication waveform as an application running on a COTS HPC server with hardware acceleration. To achieve practical data rates of today's telecommunication needs, portions of the program that are too CPU intensive would need to be off-loaded to a FPGA-based acceleration card. The OpenCL program can be partitioned into the main host program running on the CPU, called the "host code," while portions of the programs to be hardware accelerated become the optimized "kernel code." The OpenCL API is used by the host to setup kernel tasks and transfer data between the host and the hardware acceleration device(s).

For a telecommunication waveform to be implemented in OpenCL, the waveform must be properly profiled to identify what portion will be implemented as part of the host program and what portions will be implemented as the kernel to be accelerated by the hardware. Furthermore, an efficiently architected and coded OpenCL program will take advantage of the data-based and task-based parallelism capabilities for optimization.

The OpenCL framework provides some key advantages over other implementation approaches. Being a programming language, a waveform implementation can be easily ported from one computing platform to another. Although an OpenCL program is abstracted enough to offer this level of portability, it can also accommodate performance comparable to custom "purpose-built" hardware. Also, the abstraction offered by OpenCL makes it ideal for maintaining code and enhances design re-use. The OpenCL specification is occasionally expanded and revised to support new capabilities and features of CPUs and devices as they become available. For example, the OpenCL specification was updated to support direct memory access between an OpenCL acceleration device and the CPU for more efficient data transfers. This eliminated the latency and overhead of copying data in and out of the OpenCL computing device for processing, which enhanced performance. Currently, efforts are underway to integrate FPGA technology directly within the CPU core. Thus, the technological trend is for tighter integration and coupling of the FPGA technology with the general processor.

OpenCL, when combined with the host channel interface, allows one to streamline the processing of data throughout the HPC fabric. While one may code in OpenCL, to fully realize the true high-speed computing benefits, one most know how to "turn the knobs," to achieve optimal performance.

The OpenCL code is partitioned based on the Host (CPU) and Kernel (FPGA) code as shown in Figure 2. It also shows the two different compilers used to compile the host code and the optimized kernel code. One uses the host functions for items requiring much less compute-intensive functions such as routing, encapsulation/framing, management, etc. Conversely, one uses the Kernel code for logic intensive operations such as BCH encoder/decoder, Low-Density Parity Check (LDPC), symbol mapper/demapper, shaping filter, etc. The combined Host and Kernel code provide a logical partition of the various operations based on a directive from the software engineer developing the virtualized waveform. Using the Host Channels, one uses optimal operations within the HPC to support real-time and continuous data processing.

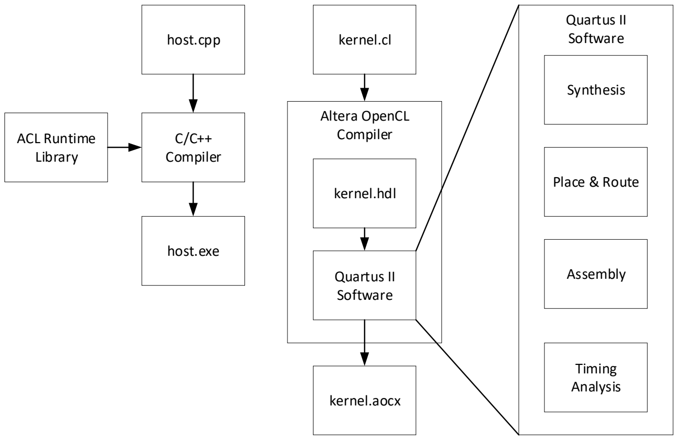

Fig. 2. HPC Hardware Architecture

IV. COMBINING THE HPC AND OPENCL

The work completed to date utilizes the OpenCL framework with a COTS Intel x86 server architecture deploying a COTS OpenCL supported FPGA-acceleration card as the computing platform for the implementation of any (satellite, tactical radio, terrestrial radio, or cellular) communication waveform. This framework offers a solution that is not dependent on custom "purpose-built" hardware. Instead it is at an abstraction layer that can be easily ported across platforms while still maintaining the performance of custom hardware.

To demonstrate the feasibility and capability of this proposed solution, the DVB-S2 waveform has been implemented in OpenCL and executed on a COTS server HPC.

Figure 3 shows the DVB-S2 Modulator and Figure 4 shows the DVB-S2 Demodulator, where each block has been implemented as an OpenCL virtualized application. The host program will be responsible for setting up and coordinating the kernel tasks, as well as, the transfer of input/output data for each function. Task-based and Data-Based parallelism will be leveraged in coding to optimize performance.

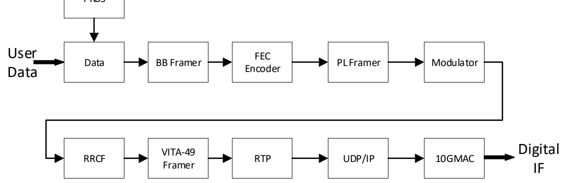

Fig. 3. DVB-S2 Modulator Blocks (Kernels)

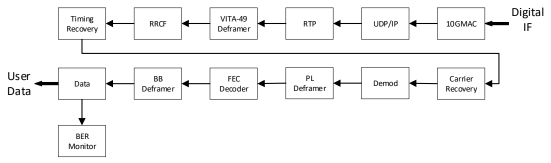

Fig. 4. DVB-S2 Demodulator Blocks (Kernels)

Each block shown in Figures 3 and 4, demonstrate an application core that is generated in OpenCL to provide the function as outlined in each functional block. The waveform engineer must have a thorough understanding of how to place the module into the proper mode section of the HPC to optimally operate based on the type of operation being performed.

V. REALIZED PERFORMANCE

To demonstrate the realized performance of the DVB-S2 waveform being supported by an HPC running OpenCL, results of a virtualized Low-Density Parity Check (LDCP) module were compared against a Matlab model. The data is presented in Figure 5. The OpenCL results were collected by streaming a signal with Additive White Gaussian Noise (AWGN) and measuring Bit Error Rate (BER) after the Decoder. As can be observed in the BER curves, better results can be achieved by optimizing the number of LDPC iterations in the OpenCL code. The trade-off is reduction in throughput as the number of iterations is increased. The results also confirmed that the BER results for the OpenCL kernel code implementation of the Decoder matched the Matlab model.

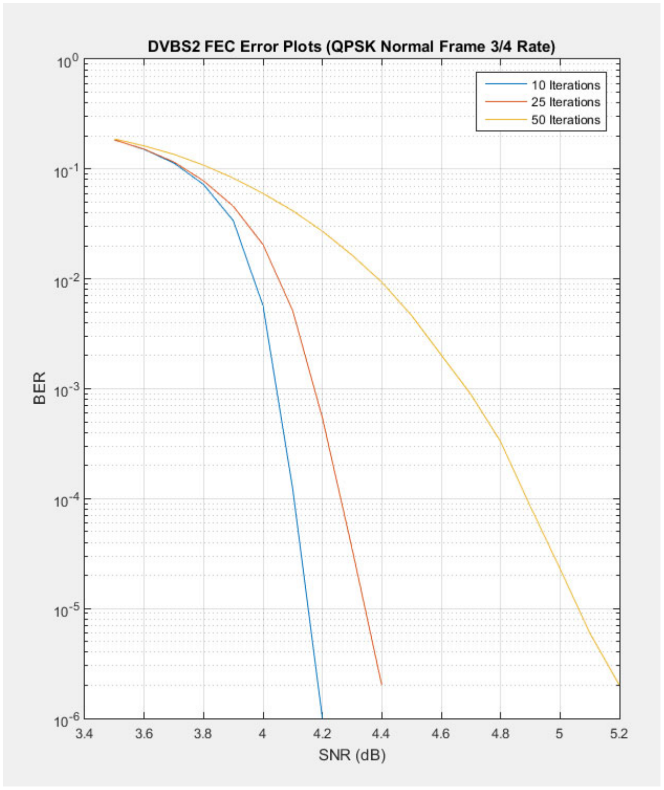

Fig. 5. DVB-S2 BER vs SNR with Multiple Iterations

For the OpenCL LDPC Decoder kernel the F_{max} measured was 191.83MHz. The performance achieved on the HPC demonstrated that hundreds of Mbps of processing could be realized. The throughput performance achieved with the virtualized core was comparable to implementations of the same functionality in low-level hardware description languages, such Verilog or VHD, running on purpose-built hardware.

VI. Moving to the Cloud

The work accomplished to date to produce a fully operational DVB-S2 waveform optimized as an OpenCL kernel developed were compared to those functions implemented on purpose-built hardware using Register Transfer Language (RTL). The throughput, maximum clock frequency, and FPGA resource efficiency were comparable to purpose built hardware implementations using RTL. This demonstrated, once and for all, that virtualization of algorithms and communication waveforms in OpenCL targeting HPCs is a viable option for an all software-based digital modems.

Such implementations could then also be executed as cloud-based applications at cloud computing centers' HPCs. Conversely, the solution could be deployed into HPC processors at the end teleport/transmission center as a point of use Modem technology.

The cloud computing resources of vendors such as Amazon Web Services (AWS), Google Cloud Computing, OVH, Microsoft Azure, etc. show a trend toward computing resources that are becoming increasingly available for hosting all software virtualized modem technology. This includes high-speed connectivity over the network to the user, as well as, the teleport where the Edge Device provides the conversion to RF signal.

The IP created by the authors have secured the concept of communication waveform virtualization within the HPC (as a high-level language) and within the cloud running on an HPC to ensure no one waveform company can prevent any other waveforms from being virtualized, thus ensuring all communication equipment providers may share in the benefits of the virtualized technology.

VII. Management and Distribtuion of Waveforms

Once a waveform has been virtualized, a control plane must be established. The control plane of a virtual modem is no different than that of a purpose-built modem. For a purpose-built modem, the typical interfaces are Web via Hypertext Transport protocol (HTTP) and Simple Network Management Protocol (SNMP). For non-cloud networks, the HPC acts as the digital modem and is assigned an IP address and the user realizes no difference in operation. Due to the need to support a secure management plane, the HTTPS (secure) and SNMPv3 (secure) are also supported.

For cloud operation, another component must be realized. The instantiation of the modem also known as "the instance" must be distributed from a library source into the point of operation in the cloud. The distribution of the instance of the virtual modem may be accomplished either as a direct download of the Modem core or requested and downloaded from an application source and remains available for the duration of the time of use.

VIII. A Solution For All Waveforms

While the groundbreaking work was accomplished for a satellite waveform, thus proving an extremely complex waveform can be supported using the HPC and OpenCL high-level language approach, the virtualization of a waveform is by no means limited to a single satellite waveform. Instead, the virtualization may be supported for many architectures, waveforms and topologies as listed below:

Satellite Access Architectures:

- Point-to-Point Single-Channel per Carrier (SCPC):
  - DVB-S2 / DVB-S2X
  - EBEM
  - etc.
- Point-to-Multipoint:
  - Frequency-Division Multiple Access (FDMA)
  - Time-Division Multiple Access (TDMA)
  - Code-Division Multiple Access (CDMA)
  - Multi-Frequency Time-Division Multiple Access (MF-TDMA)

Satellite Topologies:

- GEO
- MEO

- LEO

Tactical Radio:

- SINCGARS
- TSM-X
- DDL
- Etc.

Terrestrial Radio:

- DVB-T / DVB-T2

Cellular:

- LTE
- 4G
- 5G

IX. Digtial to Analog Conversion

While the Analog to Digital Converter (ADC) and Digital to Analog Converter (DAC) are among the most critical components in any digital communications link, their functional purpose is to act as the conversion from all digital to the analog interface (at the IF) transmit and receive edge. The simplest interface for the ADC/DAC location is generally at baseband enabling sufficient oversampling of the analog signal to optimally process in the digital domain; the fundamental constraint on this sampling process is that of the Nyquist sampling criterion, dependent primarily on the instantaneous bandwidth of the actual communications signal.

The Edge Device is completely waveform agnostic and any waveform such as SCPC, FDMA, TDMA, CDMA, MF-TDMA, etc. may be supported. The Edge Device incorporates a multi-band RF front-end with the ADC and DAC.

The digital I/Q transport protocol is optimally the FAST TIA 5041 OSDI VITA-49-compliant interface between the virtual modem and the edge device, operating over an UDP/IP encapsulated data stream carried by 1 GbE, 10 GbE, 40 GbE or even 100 GbE. The virtual modem interface allows for non-FAST TIA 5041 transport as well, since the entire modem is written in a high-level coding language, the interface for the virtual modem may be modified to support any edge device's required transport protocol.

X. Data Encapsulation (End-to-End)

The ability to carry user network traffic is the requisite function of any Modem. Since data must be carried in a manner that offers support for all data types, Generic Stream Encapsulation (GSE) is universally the encapsulation method of choice. The ability to move both standard ethernet frames and jumbo frames, which are frames larger than 1,518 bytes, through a network is required. GSE provides a minimum amount of encapsulation overhead, while supporting fragmentation of data, so that frames may be efficiently transmitted within the DVB-S2 Base-Band frame payload and reassembled at the distant end. The GSE functionality of the DVB-S2 Virtual Modem is implemented as part of the host code.

XI. TCP/IP Routing and Data Analytics

Modems of today are no longer simply providing the physical layer connectivity, but rather serve as a complete networking appliance. The HPC environment is well suited to incorporate Internet Protocols (IP) protocols, such as network routing, compression, encryption for security, and Forward Error Correction (FEC). For increased performance, any of these functions could share and leverage the integrated hardware acceleration. In other words, the same flexibility and performance that HPC with the OpenCL framework offers for the modem's physical layer implementation is also provided for the networking side as well.

Combined with Data Analytics, the virtual modem running on an HPC can now offer a real-time processing and analysis of data. This integration avoids the delay between collection of data and post-processing of data within a data center for gathering useful information and decision making.

XII. The Tactical Edge

The final topic, and possibly the most important, is the ability to take all features of the virtual modem and apply these concepts to a device deployed at the tactical edge. The tactical edge device, also referred to as the "remote device" that would be deployed out in the field at the point of use, changes the landscape of the remote modem. The remote device is the HPC architecture with the OpenCL framework implemented as a Small Weight and Power (SWaP) form factor. Prior to the virtualized modem, a purpose-built modem or modems, would be provided to the war fighter/remote user and each purpose-built modem would be used based on the desired waveform or service capability that is available in theater at any given time.

The remote device eliminates this limitation and one or a limited number of modems are required to be deployed at a given time, since each "remote device" can support as many waveforms as there are waveforms in the OpenCL library.

Therefore, a single remote device, could operate as an SCPC, FDMA, TDMA, MF-TDMA, and/or CDMA modem; or, should the need arise, the satellite modem could become a tactical radio or cellular device for a given mission, with no change in the hardware.

The remote modem may also support data analytics and general applications that may be required for a given mission. Figure 6 shows the flexibility and resiliency that can be realized using the virtual modem concept.

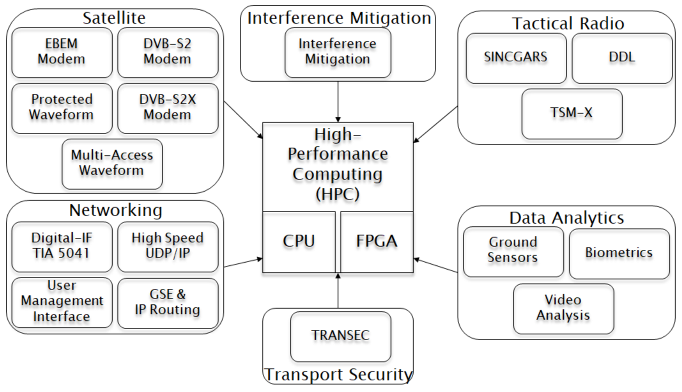

Fig. 6. OpenCL Library of Virtualized Applications

XIII. Conclusions

While this paper discusses how an SCPC waveform, DVB-S2 was successfully developed and supported by the HPC and OpenCL architecture, efforts are now underway to create additional satellite waveforms and tactical radio waveforms for SINCGARS, but the virtualization does not end there. The HPC/OpenCL environment easily lends itself to data analytics for ground sensors, video surveillance, and electronic warfare, as well interference mitigation (IM) techniques. The waveform virtualization on the HPC/OpenCL platform is only the beginning of what can be accomplished on a platform, and each supported function simply becomes another "core" that is available to the user from a library of functions. As the LEO and MEO satellite constellations continue to expand, the availability of communication options open to the user will provide global access to any communications path that is available via satellite, cellular, radio, etc., and a common waveform agnostic infrastructure will be able to support each of these communications networks.

Acknowledgment

The authors would like to thank and recognize Herald Beljour and Ben Foresta of the US Army Futures Command, CERDEC ULTR-A Lab for their guidance and support on the SBIR Phase I and II efforts, and Bill Joo of the US Navy PEO C4I PMW/A 170 S&T APM for his encouragement and guidance on supporting of Navy's future requirements.

References

[1] ETSI DVB-S2 302 307, V1.2.1, Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications (DVB-S2), August, 2009.

[2] ETSI DVB-S2 302 307-2, V1.1.1, Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 2: DVB-S2 Extensions (DVB-S2X) October, 2014

[3] ANSI TIA 5041, Future Advanced SATCOM Technologies (FAST) Open Standard Digital- If Interface (OSDI) for SATCOM Systems, April 20, 2016

[4] US Patent Number 10,177,952, Distributed processing software based modem, Michael Beeler & Kasra Toyserkani, January 8, 2019