100 Gbps Wireless - Challenges to the data link layer

Lukasz Lopacinski, Joerg Nolte
BTU Cottbus-Senftenberg
Platz der Deutschen Einheit 1
03046 Cottbus, Germany
lopacins@tu-cottbus.de
jon@informatik.tu-cottbus.de

Marcin Brzozowski, Rolf Kraemer
IHP
Im Technologiepark 25
15236 Frankfurt (Oder), Germany
{brzozowski, kraemer}@ihp-microelectronics.com

December 23, 2013

***Abstract*—The design of high speed wireless networks is a challenging task. In this paper, basic problems of an implementation of a 100 Gbps parallel data link layer processor are discussed. Such a high data rate requires a fast and low latency memory for Automatic Repeat reQuest (ARQ). Two popular memory types were investigated. Use of a DDR3 memory may lead to too long latencies, while use of a Field Programmable Gates Array (FPGA) on chip block RAM memory requires a wide memory bus and has the problem that the memory size is limited. Forward Error Correcting Codes (FEC) algorithms have to be chosen very carefully because of complexity issues. A complicated FEC may lead to huge structures and thus hardware overhead (more than 20 Virtex7 FPGAs). Even with less complicated FEC, there is probably a need to use multiple FPGAs and fast interconnect interfaces between them. For this reason high speed serial input-output transceivers are introduced. Another problem is the installation of such a network interface card in a PC system; all introduced high speed development kits did not support a bandwidth of 100 Gbps for PCI express.**

***Keywords*—Wireless, 100 Gbps, FPGA, FEC**

I. Introduction

Wireless systems are a big driver of new and challenging research directions. There are several use cases for ultra-high speed wireless links. One of the most data-rate intensive applications is video streaming. The Japanese SuperHiVision system requires a resolution of 7680 x 4320 pixels at 60 frames per second (72 Gbps) [1]. Wireless systems, which support transfer of such a stream, may be used to connect several devices: displays, cinema systems, projectors, multimedia centers, etc. In this paper, we will focus on a few challenges and we will introduce a possible architecture of such a system. To achieve 100 Gbps transmission, more than a fast physical layer (PHY) is required. Wireless transmission at that data rate may lead to a high bit error rate (BER). The data link layer is responsible for correction and/or retransmission of corrupted frames. That process must be completed in a few nanoseconds. Intensive frame retransmissions and overhead due to error correction codes may waste a significant part of the throughput. It is important to find a trade-off between complexity and performance of different algorithms. The overhead of the data link layer and other layers have to be reduced to an acceptable level. This can be achieved by using complicated algorithms and large buffers, but limitations due to hardware capacity must be considered. This paper is related to End2End100 project and cooperate with other proposed projects of the program DFG Special Priority Program 1655 (SPP1655) on "Wireless 100Gbps and beyond", e.g. the Real100G.com and Real100G.RF. This group of projects will investigate a complete wireless 100 Gbps system at ultra-high frequencies (250-330 GHz). In the End2End100 project, our main technical idea is to investigate an innovative concept for a Network Interface Card (NIC) working at 100 Gbps wireless.

II. Related work

Several research efforts have addressed to highly efficient wireless protocols. Some of the basics mechanisms are frame aggregation and acknowledgment. Those and similar topics are investigated in [2], where ARQ schemes are tested by analytical models. [3] demonstrates the effectiveness of link adaptation in the

cellular IS-856 standard. Similar estimations are done in [4] for 802.11a wireless LANs. [5] and [6] consider hybrid-ARQ schemes for wireless links. Authors in [7] presents a 60 GHz PHY and MAC demonstrator with frame aggregations and ARQ. All these papers introduce algorithms, which can be used for goodput improvements for the 100 Gbps data link layer. Other interesting techniques are forward error correcting codes (FEC) and architectures of a FEC system for ultra-high speed data links. Some authors consider packet level FEC [8], [9] for wireless transmission, but we will focus on an implementation of a block or a convolutional FEC code for 100 Gbps transmission. Some previous work is related to this topic and many papers discuss the throughput and calculational effort of such solutions [10]. Throughput limitations for FEC systems are known issue for years [11] and 100 Gbps transmission probably requires an array of processors to achieve the ultra-high target data rate. We will use some of available implementations of two popular FEC algorithms (RS, Viterbi decoder) and we will test how fast this solution can run on a modern Virtex7 FPGA (Field Programmable Gate Array) board.

Some work related to 100 Gbps wired transmissions has been published in [12] and a 100 Gbps Ethernet standard [13].

III. Challenges

Ultra-high speed wireless communication poses many challenges to the data link layer. In this section, we describe the major challenges of protocol processing and their impact on the hardware.

A. Data Link Layer

Although there are already 100G Ethernet demonstrators [14], wireless communication still cannot achieve such high data rates. The major reason is the more complex processing, mainly required to deal with higher bit error rates of wireless communication. Furthermore, wireless transmissions suffer from short coherence times, meaning the channel changes frequently. To deal with these problems, wireless transceivers apply various means such as complex baseband processing, ARQ (Automatic Repeat reQuest) or FEC (Forward Error Correction).

Automatic repeat request and forward error correction

ARQ is one of the major solutions that can deal with transmission errors [15]. That is, when the receiver does not receive a frame correctly, the transmitter will send the frame again. Obviously, ARQ may lead to performance degradation, as the transmitter waits for acknowledgments (ACK) before sending next frames. To solve that problem, the receiver sends a single ACK for several frames. This solution improves efficiency considerably [7], but requires more memory and a more complicated state machine on the TX and RX devices. In the case of ARQ use we speak of “optimistic” error correction since faults are corrected only after they have been detected by the protocol layer. Thus extra redundancy is only inserted into the transmitted packet for error detection.

Another group of solutions that deal with transmission errors are FEC algorithms. The transmitter adds extra information to the frame, and/or decodes the frame according to some schema. It allows the receiver to repair, depending on the used code several bit errors, which occurred during transmission. Clearly, this reduces the number of transmitted frames but requires extra processing on both the sender and the receiver. Furthermore, FEC increases also the frame size, since it adds extra recovery data to frames. In the case of FEC we speak of “pessimistic” error correction since the sender already incorporates redundant information into the packet that can be corrected by appropriate error decoders on the receiver side.

Both ARQ and FEC must perform extremely quickly, using only a few nanoseconds processing time for a single frame, to support 100G communication. Furthermore, 100G transceivers with ARQ need a large amount of memory to store transmitted but not acknowledged frames. We discuss these challenges later in this section.

Splitting and merging

The IEEE 802.3ba standard uses parallel lanes [16] to deal with 100 Gbps transmission. A similar approach can be adopted for 100G wireless transmission, that is, there may be several physical channels to support such high data rates. Furthermore, even if there is only a single physical channel, we may still split the data stream into several parallel lanes in order to achieve fast enough frame processing. Figure 1 shows the vision of our demonstrator platform with parallel processing. Parallel processing of the data streams requires splitting the stream into several lanes at the transmitter and merging it at the receiver. Bearing in mind that a single frame must be handled within a few nanoseconds to support 100G transmission, splitting and merging is extremely challenging for 100G wireless networks.

B. Hardware

Memory

In general, transceivers need memory to store outgoing and incoming frames. In the former case, the applica-

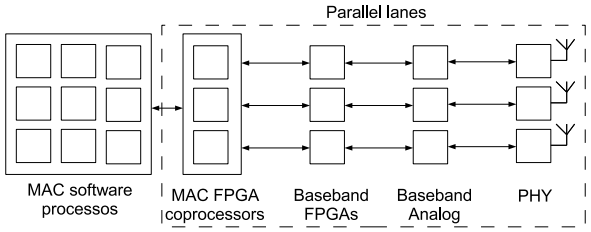

Figure 1. System architecture

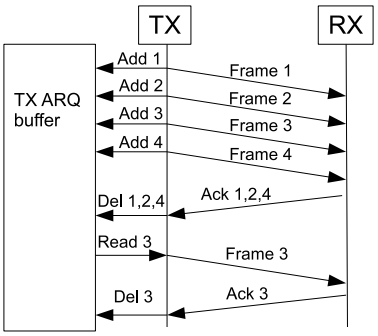

Figure 2. TX ARQ

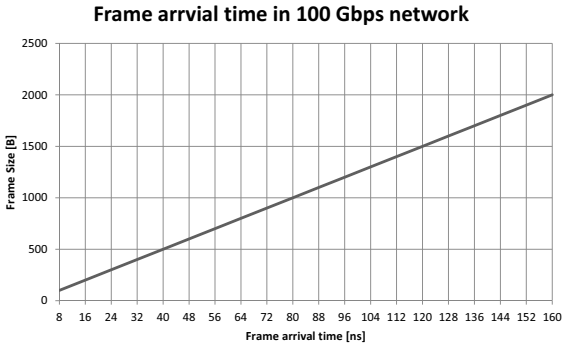

Figure 3. Frame arrival in 100 Gbps network

tion or higher-layer protocols write outgoing data to the TX buffer. In the RX path, the transceiver writes incoming frames into the RX buffer, which is often available to higher-layer protocols. Clearly, too small buffers result in congestion problems, leading either to packet losses or require some extra protocols to realize flow control. Using flow control avoid packet loss but leads to considerable performance reduction.

The problem of too small buffers is even more challenging when ARQ is applied. That is, the sender stores all outgoing frames in the buffer until the receiver sends an acknowledgment (see Figure 2). Since wireless communication suffers from a high bit error rate, we assume that ARQ and other solutions will be inevitable in 100 Gbps wireless systems to deal with transmission errors. However, ARQ with high-speed communication requires sufficient memory to store the transmitted frames. For example, at 100 Gbps data rate, the transceiver needs 12.5 GB of memory to store the frames transmitted over the last second. Similar problems arise on the receiver side, especially when the data link layer must assure the right order of incoming frames.

Although state-of-the-art computers have a few GB DDR3 memory available, such memory is too slow for 100 Gbps packet processing. The estimated access time of DDR3 memory (800MHz I/O clock) is 45ns, whereas packets in 100 Gbps networks arrive every 6.7 ns [12].

Prototyping of network cards is typically done on FPGA boards, and we will follow this idea. Such boards have limited amount of high-speed memory, needed for 100 Gbps packet processing. For example, a typical Virtex7 FPGA can buffer only about 4.5 MB of data in Block RAM memory [17]. With such an amount of memory, the TX or RX buffer will overflow after about 370 us.

The problem of limited memory in high-speed wireless communication was addressed in Ref. [18], which introduces a 60 GHz transceiver with a data rate up to 5 Gbps. The transceiver works well with a buffer size around 1 MB, and more memory does not improve the performance significantly. If we theoretically scale this system to support 100 Gbps, i.e. by factor 20, we will need about 20 times the amount of memory for buffering.

Another high-speed wireless transceiver [7], based on 60 GHz frequency, supports a data rate of 1.2 Gbps.

Processing effort

When sending or receiving a frame, both the transceiver (TX) and the receiver (RX) perform some operations. For example, the TX updates frame headers, calculates checksums, and splits data into frames and segments. To achieve the 100 Gbps data rate, each frame must be processed within a few nanoseconds, depending on the frame size (see Figure 3). For instance, TX and RX must process a frame of 500 bytes within 40 nanoseconds. Clearly, this puts an enormous stress on the processing device such as a processor or FPGA.

Nonetheless, there are already 100 Gbps Ethernet systems, which work even on state-of-the-art generic purpose processors, that is, four Intel Xeon cores without use of hardware accelerators [12]. Although such a configuration supports the 100 Gbps data rate, it dissipates as much as 650 Watts. Furthermore, since wireless communication needs more processing power than Ethernet, for instance to deal with higher bit error

rates, this would lead to even higher power dissipation. This shows clearly the need of hardware solutions, such as an FPGA or ASIC, when building a 100 Gbps wireless transceiver.

Among all operations carried out on wireless transceivers, Forward Error Correction (FEC) requires the most processing power. To outline this problem, we estimated resources needed to implement two major FEC solutions, a Viterbi decoder and Reed-Solomon code, on FPGA platforms. In the former case, we consider our Viterbi decoder implementation based on five-bit soft coding which works with a 170 MHz clock on a Virtex7 FPGA. To support 100 Gbps wireless transmission, we would need about 589 such decoders working in parallel (100 Gbps / 0.17 GHz gives 589 instances of the decoder). Our Viterbi decoder requires almost 7 million LUTs (look-up tables) and more than 10 million FFs (flip-flops) on Virtex7 FPGA. To implement all decoders needed for 100 Gbps transmission, we would need more than 20 FPGA boards in the best case, as we did not consider extra resources for routing, etc. We carried out a similar evaluation of our Reed-Solomon (RS) implementation, which limits the FPGA clock to 270 MHz and achieves 2 Gbps throughput. Thus, we would need about 100 such RS-coders, 50 TX encoders and 50 RX decoders, to support 100 Gbps transmissions. It would lead to about 10 thousand LUTs and 8 thousand FFs, which should fit into a single Virtex7 FPGA.

IV. Architecture

In previous paragraphs, some challenges for the planned activity are listed. We will now consider an example architecture of the proposed data link processor. To run at the planned data rate, most probably parallel lanes processing will be needed. One solution is to map parallel lanes to FPGAs. An example architecture with two lanes is shown in Figure 4. Two lanes will probably not be enough, but this structure is presented to keep the figure to reasonable size. The main idea is to use frames which are divided into segments, where the parts may have a dynamically changing size in view of the channel quality. The smallest retransmission unit will be the segment. Each segment may be protected by an individual FEC code, but this will depend on the channel quality. When the channel has a low Packet Error Rate (PER) and the channel quality indication is beneficial, the FEC can be disabled to increase the throughput. An ARQ scheme will provide additional robustness. The ARQ will work on segments, so that in case of frame errors only the defective segments must be retransmitted, but not the whole aggregated frame. Optionally, a FEC code can be appended to the end of the frame to protect whole structure (header and segments). To find the optimal segment and frame lengths, ARQ scheme, and FEC strategy, a Matlab simulation of the data link layer will be conducted. We will also consider some simplifications and the difference between planned and optimal settings will be estimated. It may happen that the optimal solution from the simulation will be too complicated to implement in a real demonstrator, thus there might be the need to introduce further simplifications into the planned realization.

Figure 5 presents a possible connection and separation of TX lane between the Tilera CPU board and the FPGA part. This is only one of several possible implementations of a software-hardware solution. Memory intensive operations could be realized in CPUs and implemented in C, calculation intensive operations could be calculated on FPGAs. The C implementation should speed up coding time, but the problem of memory latency must be considered. There is no other possibility to connect the FPGA with the Tilera board than a 10 Gbps Ethernet. The Tilera board has no other interfaces.

Another important aspect of a 100 Gbit demonstrator is the possibility to install in some other system. We consider a Network Interface Card (NIC) that can be installed in a PCI express (PCIe) slot in a desktop PC or a server machine. First of all, we need to estimate how fast one PCI express slot is. Such a port theoretically supports slightly more than 100 Gbps in peak throughput. There are many PC motherboards which support PCIe 3.0 x16, so there is at least one interface on a modern PC which theoretically supports such a fast NIC [19]. On the other hand, there is the need to find hardware, which is equipped with a PCIe 3.0 x16 ports. Let us consider four solutions: the Virtex FPGA VC707 and VC709 development kits, a software solution from Tilera (TILEncore-Gx72) with 72 CPU cores, and a COMBO-100G FPGA card from Inveatech. Unfortunately, none of the suggested hardware solutions currently supports the PCIe 3.0 x16 [17], [20], [21], [22]. The last solution from Inveatech could theoretically support 100 Gbps but at this time, there is only little information available, so that it currently cannot be considered as an implementation platform. The VC707 is equipped with PCIe 2.0 x8, VC709 and TILEncore-Gx72 with PCIe 3.0 x8 and this is less than 100 Gbps [19]. None of the suggested development kits can support a desktop PC with the required bandwidth. A development board should also support an interface to a baseband processor. We again consider the previously mentioned boards. The Tilera board may support up to eight 10 Gbps Ethernet ports. That is not enough for the planned implementation and there is no other interface, which can be used. Alternative FPGA development kits are equipped with up to four SFP/SFP+ ports. This is also

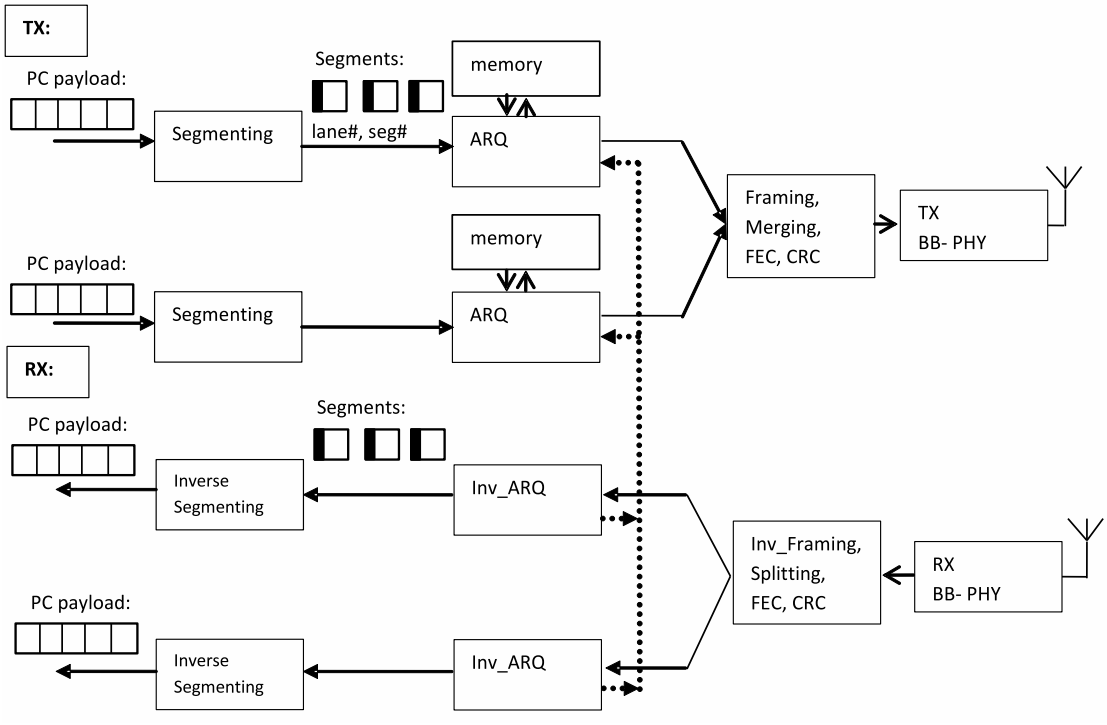

Figure 4. Lanes

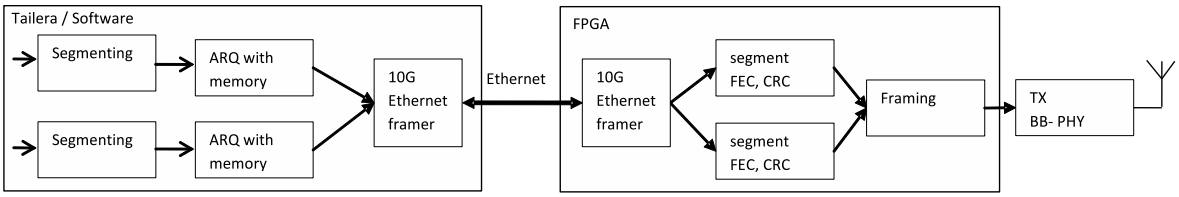

Figure 5. Mixed Software-Hardware solution

not enough to transfer 100 Gbps. The Virtex7 device has another possibility to connect to the baseband processor, namely GTX transceivers. Each transceiver can transfer up to 12.5 Gbps [17]. To achieve the requested data rate at least eight channels are needed, but VC707 and VC709 do not support so many GTX channels in the default configuration. Some FMC-HPC extensions boards are required to get access to all the on-board GTX transceivers. The best solution is to use a VC7203 kit, where up to 36 GTX transceivers are supported, but such board has no PCIe port at all [23]. There are several problems with the choice of a suitable development platform because of limited connectivity. For this reason, probably more than one of the discussed boards has to be used.

V. Conclusion

An implementation of a 100 Gbps wireless system is a challenging task. There is a problem when choosing an implementation platform since most of the available developments boards cannot achieve 100 Gbps due to interface limitations. Another problem is the FEC system. For Viterbi decoders a different solution than the current IHP implementation is needed. For example the IP core provided by Xilinx is a more promising solution (faster and smaller, but not in the scope of this paper because of licensing restrictions). Top priority is to find a solution, which requires very little resources. Fifty or more parallel lanes probably are required. Another aspect is memory performance limitations for this reason internal FPGA block RAMs are proposed. The authors plan to use multiple FPGAs connected in parallel to achieve the 100 Gbps data rate.

REFERENCES

[1] S. Namiki, T. Kurosu, K. Tanizawa, J. Kurumida, T. Hasama, H. Ishikawa, T. Nakatogawa, M. Nakamura, and K. Oyamada, "Ultrahigh-definition video transmission and extremely green optical networks for future," *Selected Topics in Quantum Electronics, IEEE Journal of*, vol. 17, no. 2, pp. 446–457, 2011.

[2] L. Badia, N. Baldo, M. Levorato, and M. Zorzi, "A markov framework for error control techniques based on selective retransmission in video transmission over wireless channels," *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 3, pp. 488–500, 2010.

[3] E. Esteves, P. J. Black, and M. I. Gurelli, "Link adaptation techniques for high-speed packet data in third generation cellular systems," in *European Wireless Conference*, 2002.

[4] D. Qiao, S. Choi, and K. G. Shin, "Goodput analysis and link adaptation for ieee 802.11 a wireless lans," *Mobile Computing, IEEE Transactions on*, vol. 1, no. 4, pp. 278–292, 2002.

[5] S. Falahati and A. Svensson, "Hybrid type-ii arq schemes with adaptive modulation systems for wireless channels," in *Vehicular Technology Conference, 1999. VTC 1999-Fall. IEEE VTS 50th*, vol. 5, pp. 2691–2695, IEEE, 1999.

[6] S. Choi and K. Shin, "A class of adaptive hybrid arq schemes for wireless links," *Vehicular Technology, IEEE Transactions on*, vol. 50, no. 3, pp. 777–790, 2001.

[7] M. Ehrig and M. Petri, "60ghz broadband mac system design for cable replacement in machine vision applications," *AEU-International Journal of Electronics and Communications*, 2013.

[8] C. Huitema, "The case for packet level fec," in *Protocols for High-Speed Networks V*, pp. 109–120, Springer, 1997.

[9] E. Ayanoglu, S. Paul, T. F. LaPorta, K. K. Sabnani, and R. D. Gitlin, "Airmail: A link-layer protocol for wireless networks," *Wireless Networks*, vol. 1, no. 1, pp. 47–60, 1995.

[10] I. Reed, R. Scholtz, T.-K. Truong, and L. Welch, "The fast decoding of reed-solomon codes using fermat theoretic transforms and continued fractions," *Information Theory, IEEE Transactions on*, vol. 24, no. 1, pp. 100–106, 1978.

[11] H. M. Ji, "An optimized processor for fast reed-solomon encoding and decoding," in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 3, pp. III–3097, IEEE, 2002.

[12] C. Hermsmeyer, H. Song, R. Schlenk, R. Gemelli, and S. Bunse, "Towards 100g packet processing: Challenges and technologies," *Bell Labs Technical Journal*, vol. 14, no. 2, pp. 57–79, 2009.

[13] O. Ishida, "40/100gbe technologies and related activities of ieee standardization," in *Optical Fiber Communication Conference*, Optical Society of America, 2009.

[14] M. Daikoku, I. Morita, H. Taga, H. Tanaka, T. Kawanishi, T. Sakamoto, T. Miyazaki, and T. Fujita, "100-gb/s dqpsk transmission experiment without otdm for 100g ethernet transport," *Journal of Lightwave Technology*, vol. 25, no. 1, pp. 139–145, 2007.

[15] R. Comroe and D. Costello Jr, "Arq schemes for data transmission in mobile radio systems," *Selected Areas in Communications, IEEE Journal on*, vol. 2, no. 4, pp. 472–481, 1984.

[16] "802.3ba media access control parameter, physical layer, and managment parameters for 40 gb/s and 100 gb/s opeariotn," 2010.

[17] Xilinx, *7 Series FPGAs Overview DS180 Advance Product Specification*, July 2013.

[18] J. Luo, A. Kortke, and W. Keusgen, "An efficient frame aggregation and block-ack scheme for 60 ghz short-range point-to-point transmission," *Wireless Personal Communications*, vol. 69, no. 1, pp. 53–73, 2013.

[19] L. Gwennap, "Sandy bridge spans generations," *Microprocessor Report*, vol. 9, no. 27, pp. 10–01, 2010.

[20] Xilinx, *UG885 VC707 EvaluationBoard for the Virtex-7 FPGA*, August 2013.

[21] Xilinx, *VC709 Evaluation Board for the Virtex-7 FPGA UG887*, June 2013.

[22] Tilera, *TILEncore-Gx72 Intelligent Application Adapter*, September 2013.

[23] *VC7203 Virtex-7 FPGA GTX Transceiver Characterization Board UG957*, July 2013.