High-Performance and Low-Cost Designs for Advanced Applications on Embedded Systems via HW/SW Co-Design and Future Interests

Weiwen Jiang

Ph.D. Candidate
College of Computer Science
Chongqing University
Chongqing, China

Visiting Scholar
Department of ECE
University of Pittsburgh
Pittsburgh, United States

Email: jiang.wwen@pitt.edu

Abstract

Recently, the research interest in deep learning is growing rapidly, which promotes the emerging of a large number of new Neural Networks (NNs). We observe two trends in the development of NNs. (1) With the objective of improving accuracy, the structures of NNs become complicated. NNs will no longer to be a simple path-structure with few layers; instead, they evolve to have complicated data dependencies and to go deeper. (2) With the fast-growing neural network markets, NNs are integrated in many real-time embedded applications, such as automotive ADAS, Aerospace, and surveillance. These applications commonly have strict requirements on high timing performance and low energy consumptions, both of which are usually less considered in the design of NNs. On top of the above observations, when it comes to system-level implementations, the design of systems with the maximum energy efficiency while satisfying the timing constraints has become a more imminent challenge. We propose to deal with the implementation challenges through the HW/SW co-design framework from the following three aspects.

First, to respond to the contradiction between the growing size on NNs and the stick real-time constraints, it is natural to scale up the size of NN accelerator (e.g., employing multi-FPGA cluster instead of using a single FPGA). The deployment problem comes along with the scale-up process; specifically, we need to decide how to partition and map NNs onto the platform.

Second, for real-life applications, the system should not only process NN tasks but also general tasks. A promising computing architecture will integrate general computing units (e.g., CPUs and GPUs) together with NN accelerators (e.g., FPGAs and ASICs). Such architecture commonly equips "cache" to accelerate memory accesses; however, it brings uncertainties (cache hit/miss) in executions. In the design phase, we need to figure out a mapping from given applications to heterogeneous computing units to maximize the energy efficiency while guaranteeing the real-time constraints with a required probability.

Last but not least, with the increasing scales and heterogeneities in the NN-oriented systems, it is hard to apply a global clock to provide a unified reference for synchronization. In consequence, computing units will be controlled by different clock references, which is known as an asynchronous system. With the consideration of asynchronous controls in NN accelerator designs, there will be unprecedented opportunities and challenges. We have presented theoretical frameworks to analyze system behavior and performance. On top of it, we plan to further devise algorithms to deploy NNs with the consideration of asynchronous controls.

I. Mapping Convolutional Neural Networks onto Heterogeneous Multi-FPGA Clusters

A. What's the problem?

Quick Answer: One FPGA cannot satisfy DNNs with real-time constraints. We propose to employ multiple heterogeneous types of FPGAs for implementing DNNs to achieve the maximum timing performance and energy efficiency.

FPGA, due to its energy efficiency, high performance, fast time-to-market and reconfigurable properties, has become one of the most popular implementation platforms for CNN inferences. Therefore, a large number of research efforts have been made on the design of FPGA-based CNN accelerators. Existing FPGA-based CNN accelerators typically employ single-FPGA architecture; however, it may fail to meet the real-time constraints that required in many embedded applications, because the limited resources on a single FPGA cannot satisfy the computation and memory requirements for CNNs. Multi-FPGA architecture is promising to boost the overall system performance by exploiting the parallelism in CNNs. Specifically, one or multiple CNN layers can be implemented on an FPGA, and a cluster of FPGAs can work together in a pipelined fashion.

The straightforward way to extend the single-FPGA design is to deploy CNNs on multiple homogeneous FPGAs. However, it diminishes the opportunities to optimize the overall performance and energy efficiency, since the computation requirement among CNN layers varies greatly. On the contrary, multiple heterogeneous FPGAs provides more design options, such that we can map CNN layers to the most suitable FPGA to improve energy efficiency. Nevertheless, how to optimally map CNNs to heterogeneous FPGAs has not been investigated. Unlike in homogeneous architectures that cost can be optimized by minimizing the number of FPGAs, *the cost together with the throughput of heterogeneous FPGAs is greatly affected by the partition and the assignment of layers*. Thus, optimal algorithms to fine-tune these configurations are essential for heterogeneous FPGAs.

B. What have we done?

Quick Answer: (1) We have built up a multi-FPGA platform using Xilinx FPGAs. (2) We have devised algorithms to optimally map CNNs onto the multi-FPGA platform to maximize throughput and minimize energy consumption[1].

- **Experimental Platform**: We have built up the experimental platform based on the Xilinx ZCU102 evaluation board.

The implementation of multi-FPGA cluster involves three aspects, including the design of (1) physical connections, (2) drivers on FPGA, and (3) NN accelerators on FPGA. First of all, for physical connections, multiple FPGAs are connected by SFP+, where the maximum speed can reach up to 40Gbps by fully using 4 SFP+ channels in ZCU102. Second, for drivers on programmable logic, our design is based on Xilinx Aurora IP core, whose latency is less than 50 cycles. Note that the Aurora-based designs will transmit data between FPGAs without passing the ARM processors (PS). Finally, for the implementation of CNNs on FPGAs, we have written the NN accelerator using Vivado HLS.

- **Research Outcomes**: Partial of the work was presented in *DAC'18* in the poster section. The work has been accepted by *CASES'18* as full paper [1], and will be presented on October 1, 2018 in ESWEEK.

In [1], we first present the motivation to employ multiple heterogeneous FPGAs for CNNs, and pose challenges in deploying CNNs onto heterogeneous FPGA architectures. Second, we formally define the bi-criteria problems on minimizing the system cost and maximizing the throughput to optimally construct heterogeneous FPGA architecture for the given CNNs. And we have proved *the NP-hardness of the optimization problem*. Third, we systematically study how to deploy CNNs to heterogeneous FPGAs and present the accurate timing models. Based on these understandings, we devise *mixed integer linear programming (MILP) formulations* and *dynamic programming based algorithms* which can obtain the optimal mappings.

C. What will we do?

Quick Answer: (1) Investigate theoretical performance-bound for the given DNNs on multi-FPGA clusters, and devise algorithms to detect and alleviate performance bottlenecks. (2) Fine-tune the structure of a given DNN according to the hardware infrastructure. (3) Develop automatic synthesis tools to deploy DNNs.

Currently, we are investigating several interesting topics on optimizing the HW/SW co-design of DNNs and NN accelerators.

1) For the deployment of existing DNNs, we will focus on analyzing the performance-bound for DNNs on multi-FPGA clusters, and devising algorithms for the performance bottleneck detection and alleviation.
2) With the consideration of the computation and communication capability provided by the hardware, we plan to fine-tune the structure of a given DNN to achieve higher performance and energy efficiency without losing accuracy.
3) We will develop an automatic synthesis tool, which can figure out the theoretical performance boundaries based on a given DNN application and the hardware resource descriptions. It can also output the optimal solutions including how to modify the DNN and how to map the modified DNN to the hardware platform.

Kindly note that the proposed approaches are applicable not only for deploying CNNs to embedded systems but also for that to cloud computing platforms.

II. Mapping Applications to Heterogeneous Platforms Satisfying Hard/Soft Real-Time Constraints

A. What's the problem?

Quick Answer: Heterogenous computing units, including CPUs, GPUs, FPGAs and ASICs, will be integrated to satisfy the growing complicated applications (such as automotive ADAS, Aerospace, surveillance and other embedded vision applications). It is important to explore how to coordinate heterogeneous computing resources to achieve the maximum energy efficiency while guaranteeing the real-time requirements, especially with the consideration of uncertainties in executions.

With the growing demand for energy efficiency, the application-specific accelerators (e.g., based on FPGAs or ASICs) become the mainstream. Although FPGAs or ASICs are superior to the general purpose computing units (e.g., CPU and GPU) in terms of energy efficiency, CPU and GPU are still indispensable parts in the whole systems. The complicated systems (e.g., Aerospace and Defense applications), commonly integrate CPUs, GPUs, FPGAs, and ASICs, which are known as "heterogeneous system". For instance, in the Xilinx FPGA platform, "Zynq UltraScale+ EG" equips Cortex-A53 cores, Cortex-R5 cores, Mali-400 MP2 GPU, and programmable logic. Given such a heterogeneous system, it is urgent to solve the problem of *how to coordinate different computing resources (or how to map an application to one type of computation resource).* In the realistic environment, the problem becomes more challenging due to the "*uncertainties*" in executions; for instance, whether a memory access hitting or missing the cache depends on the execution contexts.

The "*uncertainties*" will lead the execution times of the same task in different executions to be varied. Existing work commonly employs the worst-case execution time for each task to deal with the uncertainties, such that the overall system performance can be guaranteed; however, however, it is too pessimistic to achieve high energy efficiency, especially for the system having soft real-time constraints. In our work, we introduce the probabilistic model which can describe the execution times as random variables. *The objective is to map tasks with probabilistic execution times to heterogeneous computing resources, such that the total system cost can be minimized, while the resultant systems can satisfy the required timing performance with a high guaranteed probability.*

B. What have we done?

Quick Answer: (1) We have devised mapping algorithms for streaming applications considering the uncertainties in executions [2], [3], [4], and built up an experimental platform based on **Odroid-XU3** [5]. (2) Another evaluation platform for NNs has been built up with **NVIDIA Jetson TX2** and **Xilinx PYNQ**, which contains CPUs, GPUs, FPGAs.

- **Experimental Platform**: We have implemented benchmarks in *StreamIT* onto *Odroid-XU3* board, and also employed *NVIDIA Jetson TX2* and *Xilinx PYNQ* to compose a heterogeneous platform with CPUs, GPUs, and FPGAs.

The Odroid-XU3 board equips 4 Cortex-A15 cores, 4 Cortex-A7 cores, and 1 Mali-T628 MP6 GPU. Based on the platform and the benchmarks in StreamIT, we build up the heterogeneous platforms. We have made efforts to build up the platform: (1) we have broken applications in StreamIT to sub-tasks, and each sub-task refers to a thread; (2) we have captured the distributions of execution times for sub-tasks, which are used in analysis and modeling; (3) we have developed an automatic tool to bind threads to cores according to the mapping obtained by our algorithms.

Another experimental platform has also been built up for deploying NNs, which is composed of one *NVIDIA Jetson TX2* (CPUs+GPU) and *Xilinx PYNQ* boards (CPUs+FPGA). We have preliminary results on mapping NNs onto the platform. The results demonstrate that with the consideration of NN compression, it is possible to take advantages of heterogeneous computing resource to achieve better tradeoffs among performance, energy efficiency, and accuracy.

- **Research Outcomes**: We have presented our work in *CODES+ISSS'16*, *LCTES'17*, *ICCD'17*, *SRC@ICCAD'17*, *PhD Forum@DAC'18*. Several conference and journal papers on this topic have been published, including [2], [3], [5], [4]. In addition, we received the *Best Paper Award* in ICCD'17.

In our work, we systematically study the problem of mapping applications onto heterogeneous platforms satisfying hard/soft real-time constraints. Our contributions can be concluded as follows.

1) We present the observations on the distributions of execution times of tasks on the realistic platform, [5].
2) A probabilistic execution time model is presented, which can accurately describe the execution time of tasks, [2], [5], [4].
3) We formulate the problem to synthesize the heterogeneous platforms with the hard/soft real-time constraints, [2], [3], [4], and we prove the *NP-hardness* of the optimization problem [4].
4) Different optimization algorithms are developed to solve the optimization problem.
   a) In [2], we present *dynamic programming algorithms* for path/tree structures with hard/soft real-time constraints
   b) In [4], we present *mixed integer linear programming (MILP)* formulations and approximation algorithms for DAG-structures with hard/soft real-time constraints.
   c) In [5], we combine the *Pareto analysis and dynamic programming algorithms* to obtain optimal solutions for the application having DAG-structures with hard real-time constraints.

C. What will we do?

Quick Answer: In the future, we will (1) explore the design space of mapping DNNs onto the heterogeneous platforms; (2) devise novel mapping algorithms to achieve the optimal designs; (3) fine-tune the structure of DNNs in terms of the computing capability and communication bandwidth among heterogeneous computing units.

With the rapid development of DNN and NN accelerators, the heterogeneous platform may integrate CPUs, GPUs, FPGAs, and ASICs. Kindly note that in such platforms, the execution time of tasks on CPUs and GPUs will be various, while that on FPGAs and ASICs will be relatively fixed. Therefore, the problem becomes more interesting since we have opportunities to map tasks with high varieties to FPGAs to further increase the efficiency of the whole system. In addition, unlike the mapping of traditional applications, the mapping of DNNs onto heterogeneous platforms will face a bundle of new challenges since the tasks in DNNs may be both data-intensive and memory-intensive.

Based on the above observations, we will first explore the design space of mapping DNNs onto the evaluation platform. Then, we will devise novel mapping algorithms to maximize the energy efficiency under the soft real-time constraints. Last, we will conduct hw/sw co-design of DNNs and platforms to boost the performance and improve the energy efficiency.

III. Challenges and Opportunities in Asynchronous/Self-Timed Designs

A. What's the problem?

Quick Answer: As the systems for DNNs consistently scale up, it is hard to synchronize the whole system using one global clock. Employing multiple clock domains can increase the system performance; however, a set of fundamental problems (e.g., performance estimation and deadlock checking and avoidance) for asynchronous communication need to be studied.

With the increasing scales and heterogeneities in NN-based systems, it is hard to apply the same clock to synchronize the whole system, indicating that systems will be performed in multiple clock domains, which are called asynchronous systems or

self-timed systems. Unlike synchronous systems that all components start to be fired triggered by the same clock, self-timed systems need to employ a more complicated protocol to coordinate the behaviors of components in different clock domain. For example, the handshaking protocol is commonly employed in a self-timed system. However, it opens up new challenges in the design phase. First, without a global clock, it is hard to capture the start and finish time for different stages; therefore a matched model is needed to accurately describe system behaviors. Second, controlled by handshaking, the rate of generating output data may not be stable, complicating the throughput and performance analysis. Third, with a global clock, only one buffer is required between two stages; on the contrary, controlled by handshaking, more buffers may be required to alleviate performance degradation caused by waiting for data/requests. With the consideration of resource-limited devices (e.g., FPGAs) and the real-time requirements, it is important to minimize the number of buffers while achieving the required performance. *In consequence, theoretical analysis and optimization techniques are required for the design of self-timed systems.*

B. What have we done?

Quick Answer: We have (1) developed a cycle-accurate simulator [6] based on SystemC; (2) studied the fundamental problems in the design of asynchronous systems; and (3) devised algorithms to optimally construct asynchronous systems.

- **Research Outcomes**: We have presented our work in *RTCSA'14*, *ICA3PP'15*, *HPCC'15*, *ICESS'16*. We have published several conference and journal papers on this topic. The selected publications are listed as [6], [7], [8], [9], [10].

We have studied the basic properties of self-timed systems, including how to compute the system throughput [10], how to avoid deadlocks [8], how to insert the minimum number of buffers into the system to achieve the required performance [6], and how to map an application to a self-timed system with the maximum throughput and the minimum buffer cost [7].

C. What will we do?

Quick Answer: We will (1) explore the design space for DNNs implemented using multiple clock domains on FPGAs; (2) devise novel algorithms to efficiently partition DNNs into different clock domains to achieve the maximum performance and energy efficiency; and (3) develop automatic tools to deploy DNNs to the systems with multiple clocks.

Future work will be conducted based on two observations in implementing DNNs onto multi-FPGA clusters.

1) In the programmable logic on an FPGA, the accelerator part and the communication part are naturally performed under two clock domain. Because the communication part requires a high-frequency clock for high-speed data transmission, while the accelerator part needs a clock with relatively lower frequency due to the long data path.
2) The NN accelerator can be partitioned into different functional components, where different components can have different clock rate to achieve the maximum overall performance.

In consequence, we will first explore the design space for DNNs implemented using multiple clock domains on FPGAs. Second, we will devise novel algorithms to efficiently obtain the optimal solutions in the design space. Last, we will develop automatic tools to help designer synthesize DNNs onto the systems with multiple clock domains.

References

[1] **Weiwen Jiang**, Edwin H-M Sha, Qingfeng Zhuge, Lei Yang, Xianzhang Chen, and Jingtong Hu. Heterogeneous fpga-based cost-optimal design for timing-constrained cnns. *Accepted by International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES-ESWEEK'18) and appear at IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2018.

[2] **Weiwen Jiang**, Edwin H-M Sha, Qingfeng Zhuge, and Xianzhang Chen. Optimal functional-unit assignment and buffer placement for probabilistic pipelines. In *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS-ESWEEK'16)*, pages 13:01–13:10. ACM, 2016.

[3] **Weiwen Jiang**, Edwin H-M Sha, Xianzhang Chen, Lei Yang, Lei Zhou, and Qingfeng Zhuge. Optimal functional-unit assignment for heterogeneous systems under timing constraint. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 28(9):2567–2580, 2017.

[4] **Weiwen Jiang**, Edwin H-M Sha, Qingfeng Zhuge, Lei Yang, Hailiang Dong, and Xianzhang Chen. On the design of minimal-cost pipeline systems satisfying hard/soft real-time constraints. *Accepted by IEEE International Conference on Computer Design (ICCD'17), Obtained the **Best Paper Award** in EDA Track, and appear at IEEE Transactions on Emerging Topics in Computing (TETC)*, 2018.

[5] **Weiwen Jiang**, Edwin H-M Sha, Qingfeng Zhuge, Hailiang Dong, and Xianzhang Chen. Optimal functional unit assignment and voltage selection for pipelined mpsoc with guaranteed probability on time performance. In *Proceedings of ACM SIGPLAN / SIGBED 2017 Conference on Languages, Compilers, and Tools for Embedded Systems (LCTES'17), and also appear at ACM SIGPLAN Notices*, volume 52, pages 41–50. ACM, 2017.

[6] **Weiwen Jiang**, Edwin H-M Sha, Qingfeng Zhuge, Lei Yang, Xianzhang Chen, and Jingtong Hu. On the design of time-constrained and buffer-optimal self-timed pipelines. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2018.

[7] **Weiwen Jiang**, Edwin H-M Sha, Xianzhang Chen, Lin Wu, and Qingfeng Zhuge. Synthesizing distributed pipelining systems with timing constraints via optimal functional unit assignment and communication selection. *Journal of computational science (JOCS)*, 26:332–343, 2018.

[8] **Weiwen Jiang**, Qingfeng Zhuge, Xianzhang Chen, Lei Yang, Juan Yi, and Edwin H-M Sha. Properties of self-timed ring architectures for deadlock-free and consistent configuration reaching maximum throughput. *Journal of Signal Processing Systems (JSPS)*, 84(1):123–137, 2016.

[9] **Weiwen Jiang**, Edwin H-M Sha, Xianzhang Chen, Qingfeng Zhuge, and Lin Wu. Optimal functional assignment and communication selection under timing constraint for self-timed pipelines. In *International Conference on Embedded Software and Systems (ICESS)*, pages 87–92. IEEE, 2016.

[10] **Weiwen Jiang**, Qingfeng Zhuge, Juan Yi, Lei Yang, and Edwin H-M Sha. On self-timed ring for consistent mapping and maximum throughput. In *International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–9. IEEE, 2014.