

RESEARCH REVIEW 2023

**Carnegie
Mellon
University**
Software
Engineering
Institute

Co-Design for Edge Artificial Intelligence – Application Specific System on Chip

NOVEMBER 8, 2023

Dr. John G. Wohlbiel
Senior Research Scientist
Advanced Computing Lab Lead

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

©2023

Document Markings

Copyright 2023 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM23-1069

Agenda

- Application-specific system on chips (SoCs)
- Accelerator design flow
- SoC integration
- Prospects
- Resources

Co-Design for Edge Artificial Intelligence—Accelerator Intellectual Property

Application-Specific System on Chips (SoCs)

Application-Specific SoCs

<https://all3dp.com/2/raspberry-pi-drone-simply-explained/>



<https://microcontrollerslab.com/system-on-chip-soc-introduction/>

- Edge devices, such as drones, are powered by microelectronics.
 - Accelerometer
 - Camera
 - Flight controller
 - GPS module
 - Speed controller
 - Radios (Tx/Rx)
- What is an SoC?
 - A chip containing components that comprise a system
 - Components: CPU cores, GPU cores, memory cores, accelerators
 - Accelerators: FFT, NLP, NVDLA, TX/RX

Application-Specific SoCs

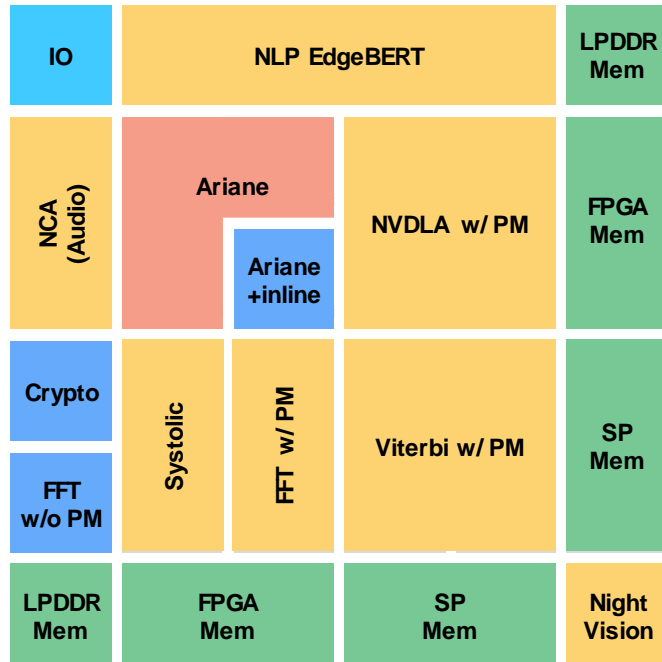


Image courtesy of IBM EPOCHS

- Electronics efficiency plays a major role in system performance.
 - Designing an SoC for a particular deployment can yield major benefit.
- What makes an SoC application specific?
 - The mix of system components is designed from the application.
 - E.g., object detection, object classification, signal characterization
- How do you design an application-specific SoC?
 - Hardware description languages (HDL): Chisel, Verilog
 - High-level synthesis (HLS): Bambu, oneAPI, AMD/Xilinx
 - Frameworks: Chipyard, Embedded Scalable Platforms

“In general, **compute** can improve mission time and lower energy consumption by as much as 5X.”

Boroujerdian, Behzad; Genc, Hasan; Krishan, Srivatsan; Faust, Aleksandra; & Reddi, Vijay Janapa. Why Compute Matters for UAV Energy Efficiency? *International Symposium on Aerial Robotics*. 2018.

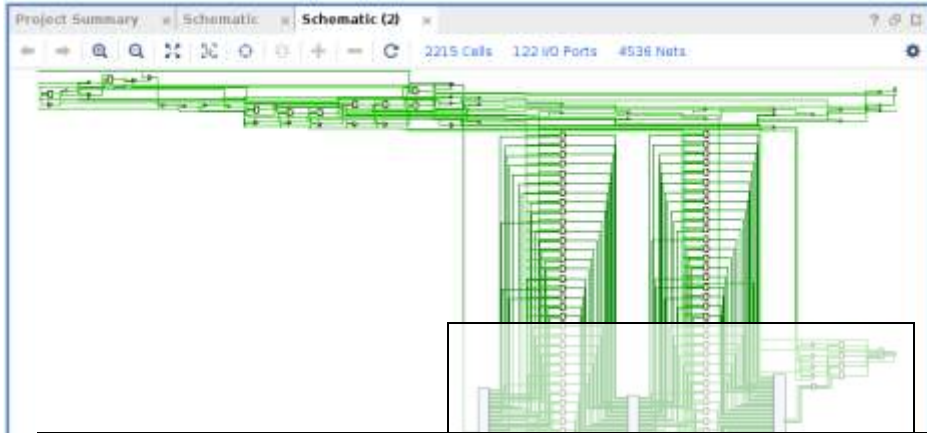
Co-Design for Edge Artificial Intelligence—Accelerator Intellectual Property

Accelerator Design Flow

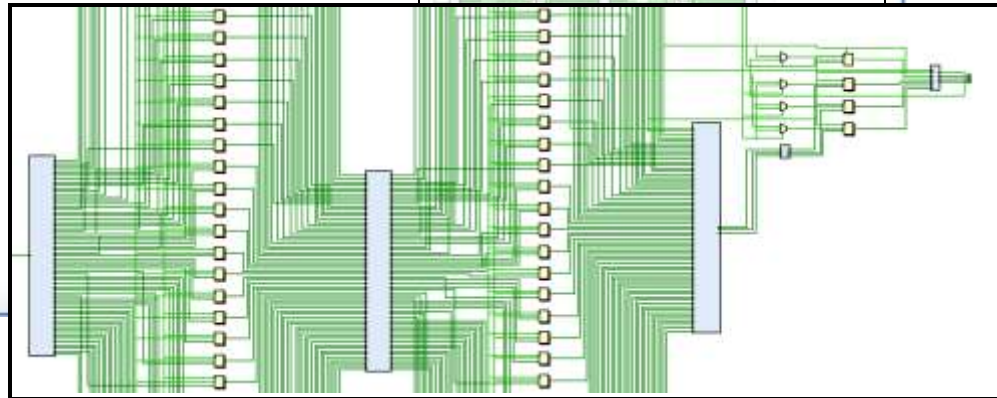
Accelerator Design Flow

- Identify hot spots of application(s) to accelerate.
 - Profiling, source code analysis, expert knowledge
- Accelerator design options
 - HDL: code the accelerator by hand (time consuming, but can achieve optimal performance)
 - HLS: tools lower code to HDL (much faster, but unlikely to achieve optimal performance)
- High-level synthesis
 - HLS tools supporting PyTorch and Tensorflow are immature.
- Emerging approaches
 - Multi-Level Intermediate Representation (MLIR) compiler framework for model lowering
 - Dialect abstraction to enable different types of optimizations

Example



- Three-layer neural network implemented in PyTorch—[PyTorch Iris](#)
 - Linear/relu/linear/relu/softmax
 - Use [hls4ml](#) with AMD/Xilinx backend for synthesis
 - Fixed point quantization: 16 bits, 6 bits left of decimal
 - HLS can be thought of as “code to circuit”
 - Ready for SoC integration



```
class Model(nn.Module):
    def __init__(self, input_dim):
        super(Model, self).__init__()
        self.relu = nn.ReLU()
        self.layer1 = nn.Linear(input_dim, 64)
        self.layer2 = nn.Linear(64, 3)
        self.softmax = nn.Softmax(dim=1)

    def forward(self, x):
        x = self.relu(self.layer1(x))
        x = self.relu(self.layer2(x))
        x = self.softmax(x)
        return x
```

Co-Design for Edge Artificial Intelligence—Accelerator Intellectual Property

SoC Integration

SoC Integration



- Using [Embedded Scalable Platforms](#) from Columbia University
 - Includes selectable CPUs, memory cores, I/O cores, selection of accelerators
- Example: four-tile design with an RISC-V CPU core, memory tile, I/O tile, and PyTorch-Iris accelerator
- Implementation on field-programmable gate array (FPGA) with AMD/Xilinx tools
 - Design flow for application-specific integrated circuit (ASIC) also available

SoC Testing—FPGA Implementation



FPGA Dev Board Xilinx VCU118 connected to host. Booting Linux on RISC-V core and then running application, invoking the accelerator.

```
gjeschbier -- root@ta-ginkbird: /home/gjeschbier -- ssh -Y ta-ginkbird.ad.ssf.cmu.edu -- 80x35

Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Aug 13 2017, 15:25:34.
Port /dev/ttyUSB1, 14:00:00

Press CTRL-A Z for help on special keys

CTRL-A Z for help | 38400 Bn1 | NOR | Minicom 2.7.1 | VT102 | Offline | ttyUSB1
```

Co-Design for Edge Artificial Intelligence—Accelerator Intellectual Property

Prospects

Prospects



- Quantification of performance with real neural networks
- Accelerators for application with digital signal processing and neural network loads
- Integrated simulation of system deployment accounting for energy savings

<https://paperswithcode.com/dataset/airsim>

Resources

- Reproduce these results using open source code
- CMU-SEI Github
 - PyTorch-Iris – hls4ml branch – simple neural network code
 - <https://github.com/cmu-sei/pytorch-iris>
 - Docker environment for using [hls4ml](#)
 - <https://github.com/cmu-sei/hls4ml-docker>
 - Docker environment for using [esp](#)
 - <https://github.com/cmu-sei/esp-docker>
- ESP tutorials
 - <https://www.esp.cs.columbia.edu/docs/>

Contact



Dr. John G. Wohlber
Senior Research Scientist
Advanced Computing Lab Lead

Telephone: +1 412.268.5800

Email: info@sei.cmu.edu