

# ARTIFICIAL NEURAL NETWORK AND ACCELERATOR CO-DESIGN USING EVOLUTIONARY ALGORITHMS

PHILIP COLANGELO<sup>1,3</sup>, OREN SEGAL<sup>2</sup>, ALEX SPEICHER<sup>2</sup>, MARTIN MARGALA<sup>1</sup>

<sup>1</sup> UNIVERSITY OF MASSACHUSETTS LOWELL

<sup>2</sup> HOFSTRA UNIVERSITY

<sup>3</sup> INTEL

# LEGAL DISCLAIMER

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

The code names presented in this document are only for use by Intel to identify products, technologies, or services in development, that have not been made commercially available to the public, i.e., announced, launched or shipped. They are not "commercial" names for products or services and are not intended to function as trademarks.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at <http://www.intel.com/design/literature.htm>.

Intel is a trademark of Intel Corporation in the US and other countries.

Copyright © 2015 Intel Corporation. All rights reserved.

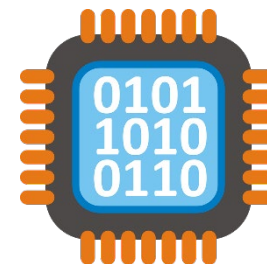
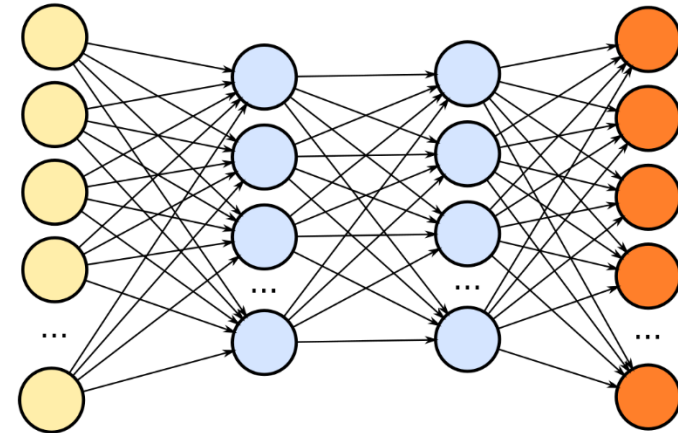
\* Other brands and names may be claimed as the property of others.

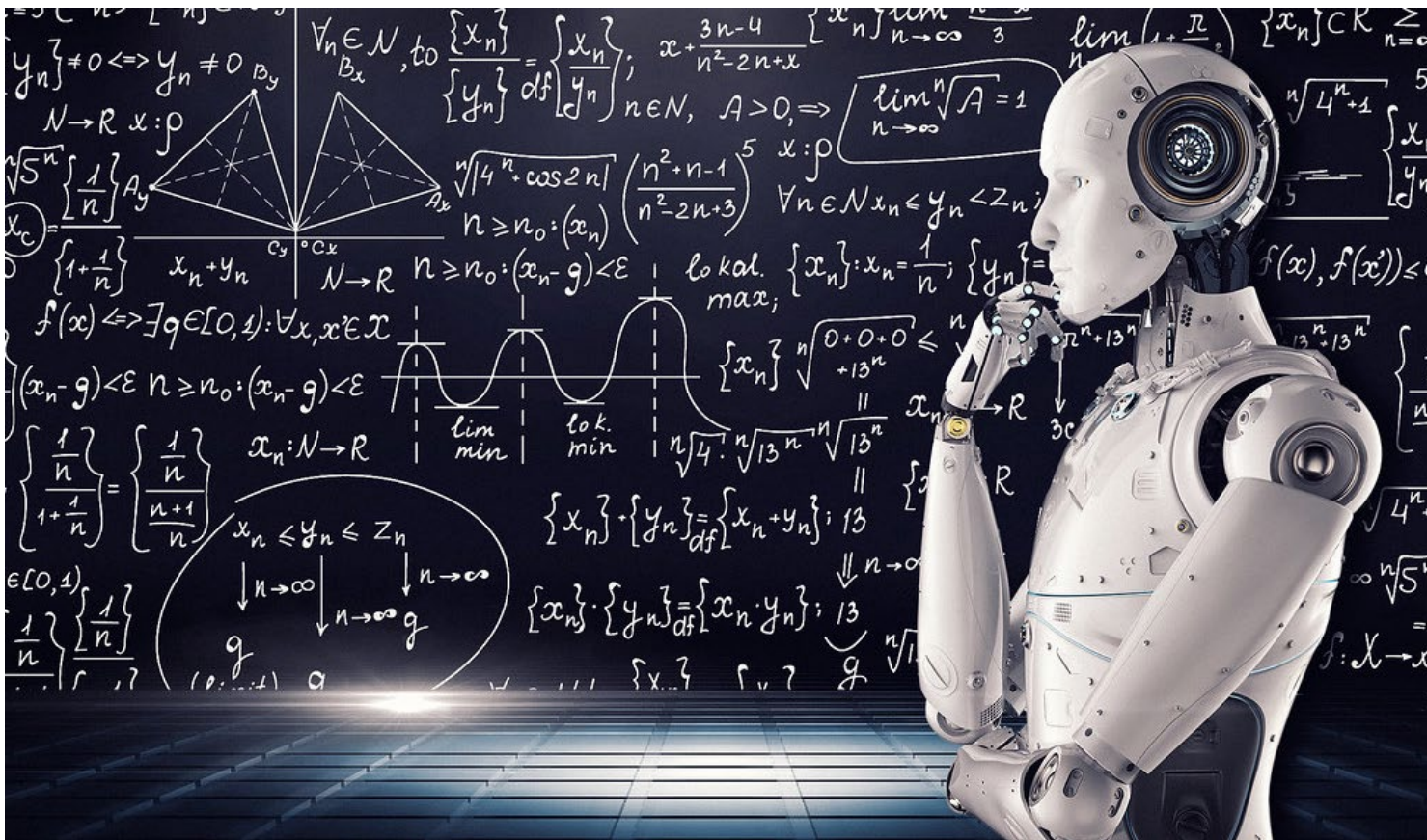
# OUTLINE

1. Introduction
2. ECAD Software
3. ECAD Hardware
4. Experiments and Conclusions

# PROBLEM STATEMENT

- Designing performant neural networks is difficult
  - Many hyperparameters to consider
  - It is not an exact science
- Fitting optimized neural networks onto hardware is difficult
  - General purpose not difficult and not optimal
  - Creating specialized hardware for a neural network is difficult
    - It is also mostly not feasible



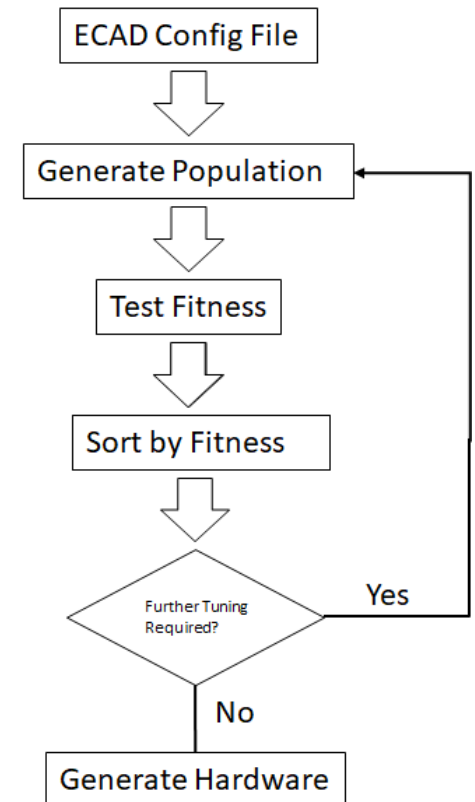


# CURRENT SOLUTIONS

- **Artificial Neural Network Architecture Search (NAS)**
  - Random search, evolutionary, reinforcement learning, Bayesian optimizations
- **Automated Machine Learning (AutoML)**
  - Make machine learning easy for non experts
  - Faster solutions
  - Better solutions
- But these methods still come with problems of their own...
- Few consider acceleration

# EVOLUTIONARY CELL AIDED DESIGN (ECAD)

- Artificial neural network and hardware co-optimization
- Configuration file
  - Initial/Base Design
  - Parameters to control the hardware target, objectives and flow
- Evolutionary based design process
  1. Generate Population
  2. Test fitness
  3. Sort by fitness
    - Repeat steps 1-3 until objectives are met
    - Generate hardware



# ECAD SOFTWARE

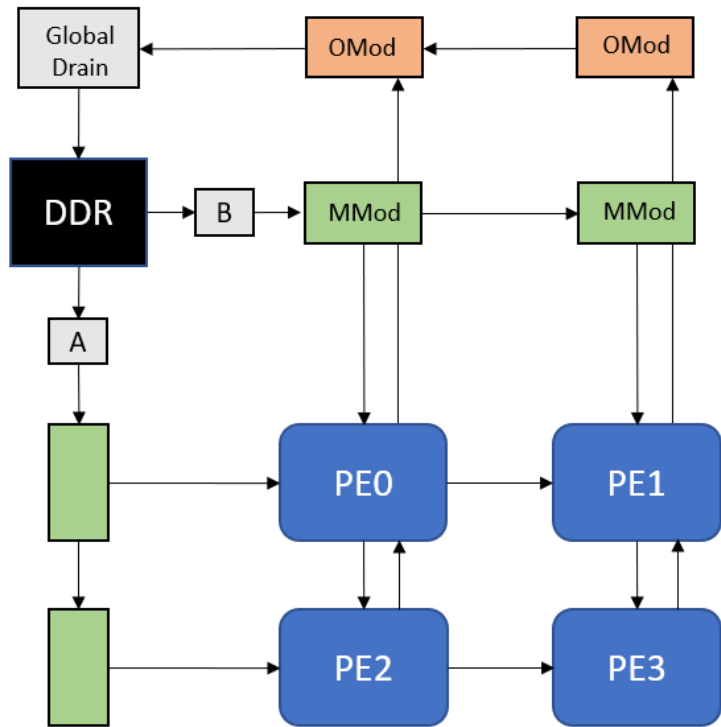
- Auto Generating NN Designs
  - Initially based on configuration file
  - Evolve designs according to fitness (repeatedly apply mutations to best performers)
- Evaluating Population Fitness
  - Simulation Worker capable of simulating a NN design
  - Physical Worker capable of synthesizing a NN design
  - HWDB Worker capable of accurately estimating synthesis results
  - Multi-objective optimization based on the specified goals (accuracy/throughput etc.)
  - Parallel implementation based on MPI sends NN models to workers for evaluation

# ECAD HARDWARE FLOW



- ECAD configuration file provides input for accelerator description
  - Memory banks (capacity, bandwidth)
  - FPGA (ALMs, M20K, DSP,  $F_{max}$ )
- 3 levels for hardware search
  - Software model for fast search, limited resource info
  - Partial compilation, minutes with better resource estimates
  - Full compilation, takes hours for results
- Outputs fitness of permutation
  - Up to the user to provide important metrics





# ECAD HARDWARE DESIGN

- Leverages reconfigurability of FPGA
- 2D systolic array
  - Pipelined data flow
  - Scalable and modular
  - Accurate modeling
- Evolving parameters
  - Rows, columns, vector width, interleaving, and scaling
- Hardware fitness
  - Resource utilization, latency, img/s, effective operations per second.

# EXPERIMENTS

- Arria 10 GX 1150 FPGA
- Four optimizations considered
  - Accuracy, latency, img/s, and effective gigaoperations/s
- Hardware only search
  - Insights into convergence of hardware designs given optimization pressure
- Hardware and accuracy search
  - Ultimate goal

# MODEL ACCURACY

- First step was to verify the model
- Top table shows full compiled results
- Bottom table shows the model results

TABLE III  
MEASURED HARDWARE PERFORMANCE RESULTS

HW Config.	Time (ms)	ALM	SRAM	DSP	$F_{max}$ (MHz)
2,8,16,16,2	9.64	37%	36%	26%	220
2,8,32,16,2	5.53	50%	51%	43%	212
4,8,8,16,18	4.65	38%	34%	26%	228

TABLE IV  
MODELED HARDWARE PERFORMANCE RESULTS

HW Config.	Time (ms)	ALM	SRAM	DSP	$F_{max}$ (MHz)
2,8,16,16,2	9.59	47%	38%	26%	220
2,8,32,16,2	4.64	61%	55%	43%	212
4,8,8,16,18	4.66	45%	37%	26%	228

TABLE I  
EA RESULTS FOR HARDWARE ONLY OPTIMIZATIONS USING SINGLE AND DUAL BANK DDR MEMORY

Optimization	# Banks	Batch Size	Neurons	BW bound	Latency (ms)	Img/s	EGOP/s	Rows	Cols	Vec	Interleave	Scale
Img/s	1	992	48; 180	Yes	0.068352	1,243,182.00	119	4	8	16	8	2
Latency	1	2	18; 142	No	0.01792	111,607.00	4.0375	2	4	8	4	2
EGOP/s	1	958	992; 1010	No	3.6495	47,508.98	170	4	16	8	16	30
Img/s	2	992	62; 190	No	0.06348	1,532,832.00	191	4	8	16	8	4
Latency	2	158	2; 64	No	0.008096	505,425.00	2.36	2	2	4	2	2
EGOP/s	2	960	998; 1014	No	2.765	49,792.86	179	4	16	8	16	32

## HARDWARE ONLY SEARCH

- Img/s provide highest performing throughput designs
- EA resulted in smaller hardware designs, 2 DDR banks had best result
- EGOP/s results in better utilization of resources

TABLE II  
TOP PERMUTATIONS FROM OPTIMIZING ACCURACY AND IMG/S

Top Permutation	Fitness Sum	Accuracy	Batch size	Neurons	HW Config.	Latency (ms)	Img/s	EGOP/s
Fitness Sum	1.065	0.936	508	852	2,8,32,16,2	0.352	129,056	174.61
Accuracy	1.006	0.942	484	1018	2,8,16,16,2	0.583	57,296	92.62
Latency (ms)	1.065	0.936	508	852	2,8,32,16,2	0.352	129,056	174.61
Img/s	1.064	0.935	572	970	2,16,32,32,2	1.283	129,144	198.93
EGOP/s	1.062	0.933	572	980	2,16,32,32,2	1.283	129,144	200.98

## HARDWARE AND ACCURACY SEARCH

- Fitness sum provides the weighted accumulation of accuracy and img/s
- Fitness sum and latency arrived at the same design
  - Table I showed Latency reducing number of neurons so with accuracy combating this, it resulted in a good design

# THANK YOU