Deploying Deep Learning on Embedded Devices – When FPGAs Make Sense

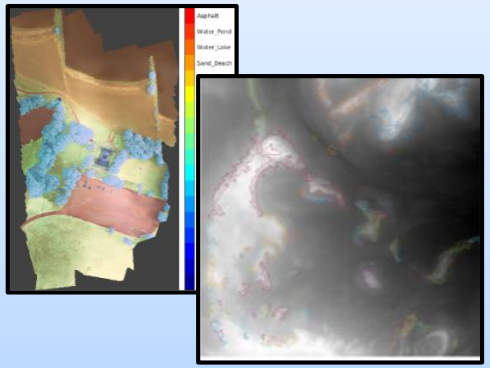
Jack Erickson

HDL Technical Marketing

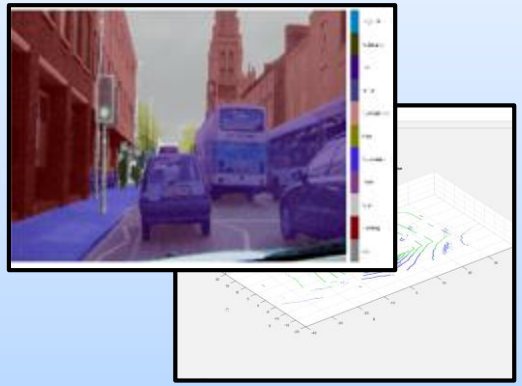
MathWorks®

Deep Learning Inferencing on Embedded Devices

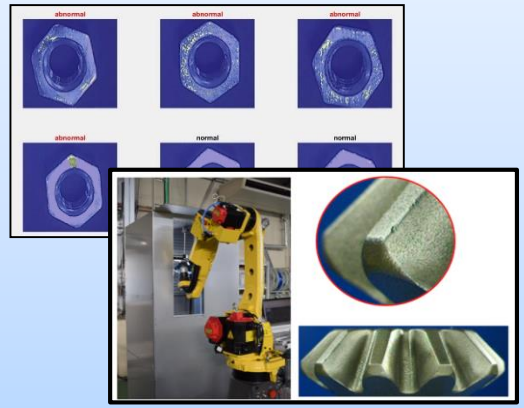
Airborne Image Analysis



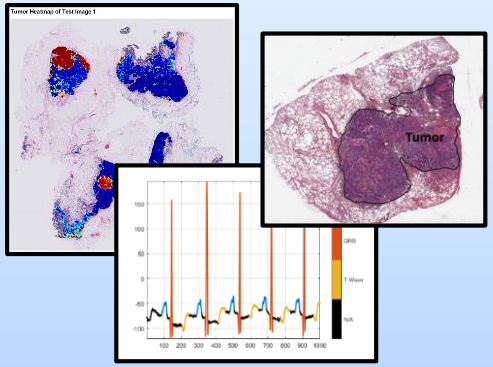
Autonomous Driving



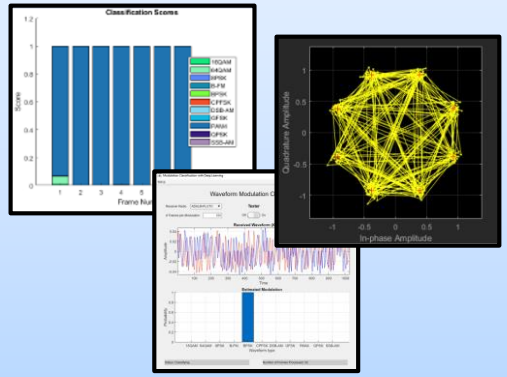
Industrial Inspection



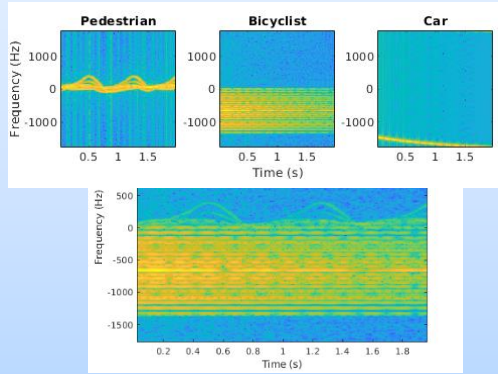
Medical Image Analysis



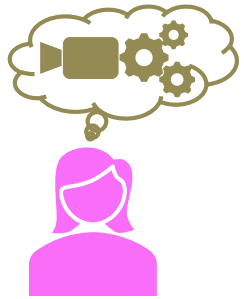
Wireless Modulation Classification



Radar Signature Classification



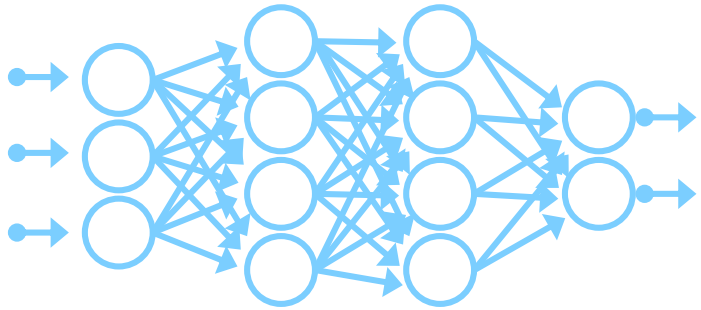
System Requirements Drive Network Design

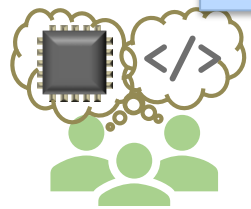


Systems Engineer

Camera specs
Accuracy
Latency
Cost
Power

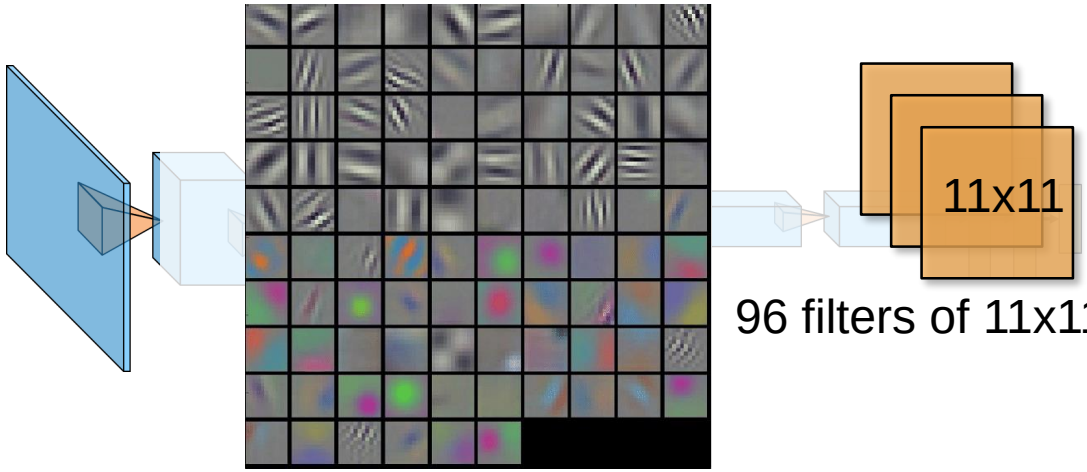
Deep Learning Practitioner



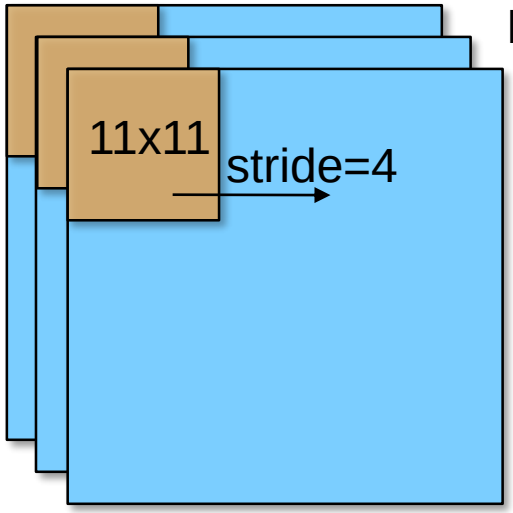


Hardware/Software Engineers

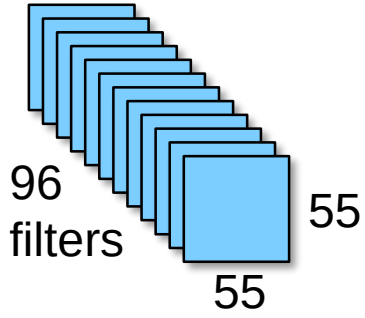
Challenges of Deploying Deep Learning to FPGA Hardware: Convolution



96 filters of 11x11x3 of 32-bit parameters →**140k bytes**



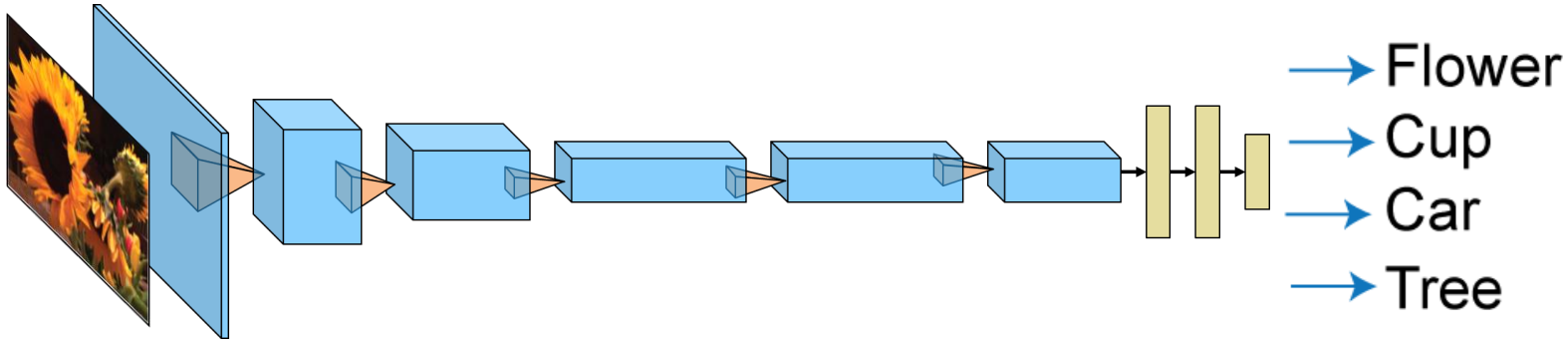
Each stride is an 11x11x3 matrix multiply-accumulate



→**1.16M bytes of activations**

→**105M** floating-point multiply operations!

Challenges of Deploying Deep Learning to FPGA Hardware



	input	conv 1	conv 2	conv 3	conv 4	conv 5	fc6	fc7	fc8	Total
Parameters (Bytes)	n/a	140K	1.2M	3.5M	5.2M	1.8M	148M	64M	16M	**230 M**
Activations (Bytes)	588K	1.1M	728K	252K	252K	168K	16K	16K	4K	**3.1 M**
FLOPs	n/a	105M	223M	149M	112M	74M	37M	16M	4M	**720 M**

→ Off-chip RAM

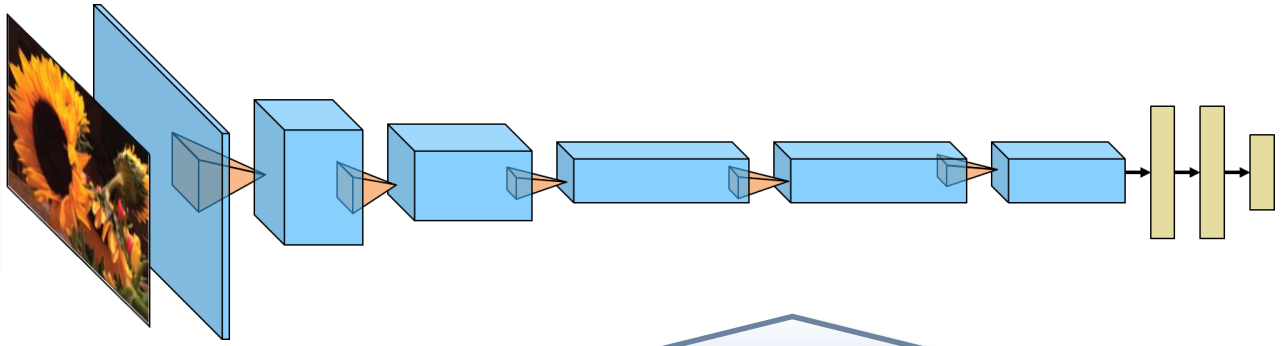
→ Block RAM

→ DSP Slices

Deploying Deep Learning to FPGA Hardware Requires Collaboration

Acquire data

Resize



Output / display

Mem i/f

	input	conv 1	conv 2	conv 3	conv 4	conv 5	fc6	fc7	fc8	Total
Parameters (Bytes)										
Activations (Bytes)										
FLOPs										

Optimize

- Network / layers
- Fixed-point quantization
- Processor micro-architecture

AI-Driven System Design

Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability

System Design

- Integration with complex systems
- System simulation
- System verification and validation

Deployment

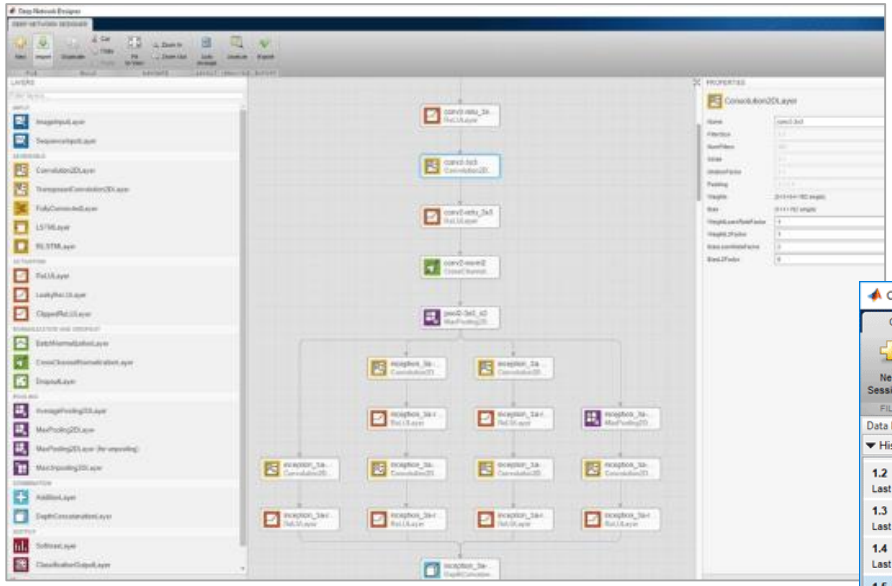
- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

Iteration and Refinement

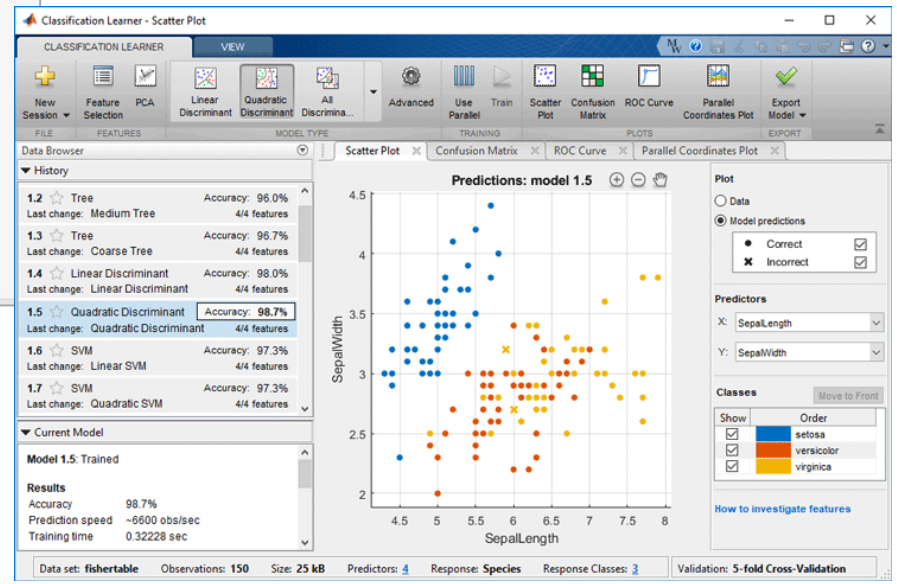
Design and Analyze Your Networks in MATLAB

AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability



Deep Network Designer app to build, visualize, and edit deep learning networks

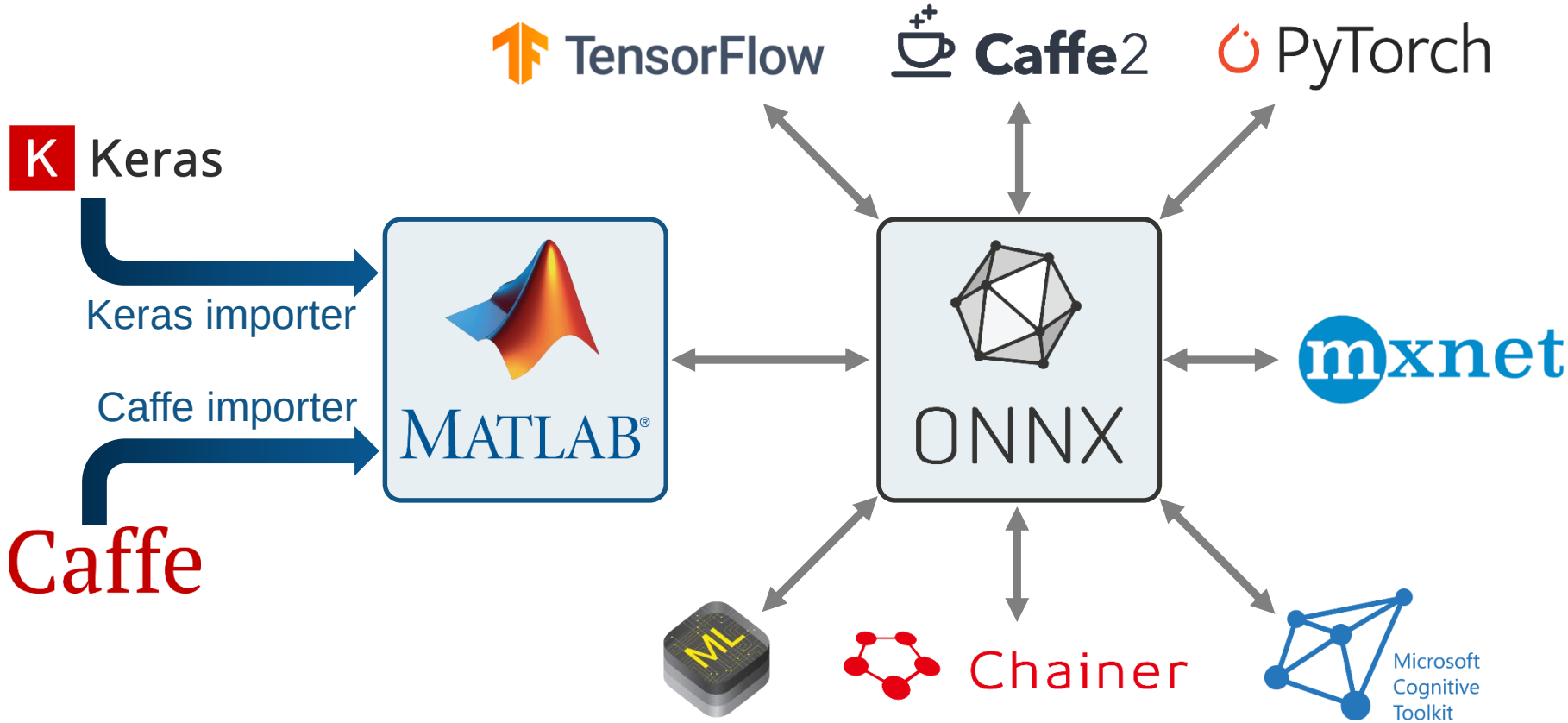


Classification Learner app to try different classifiers and find the best fit for your data set

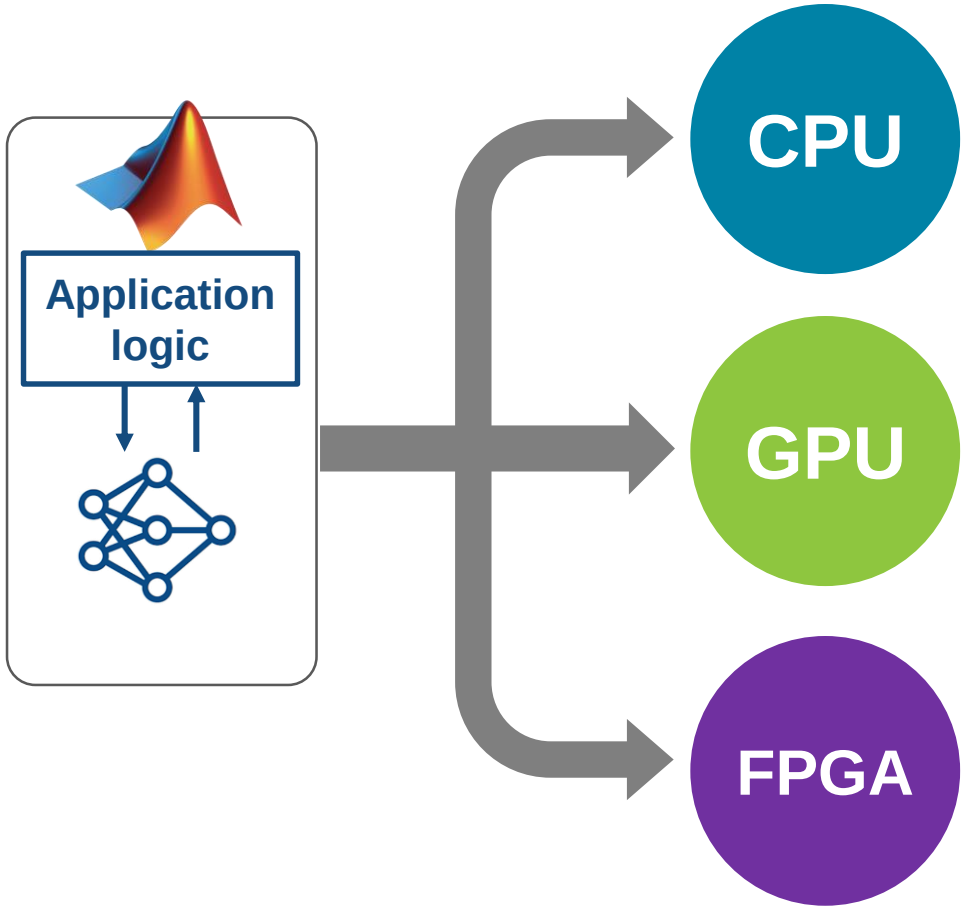
MATLAB Interoperates with Other AI Frameworks

AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability



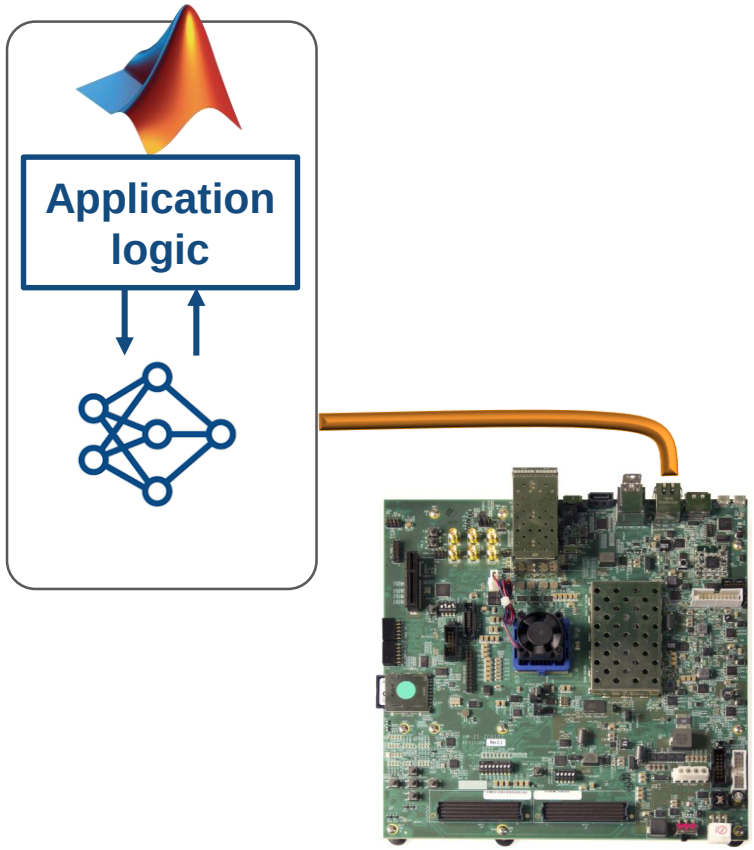
Deploy from MATLAB to a Variety of Hardware Platforms



Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

FPGA Deployment from MATLAB

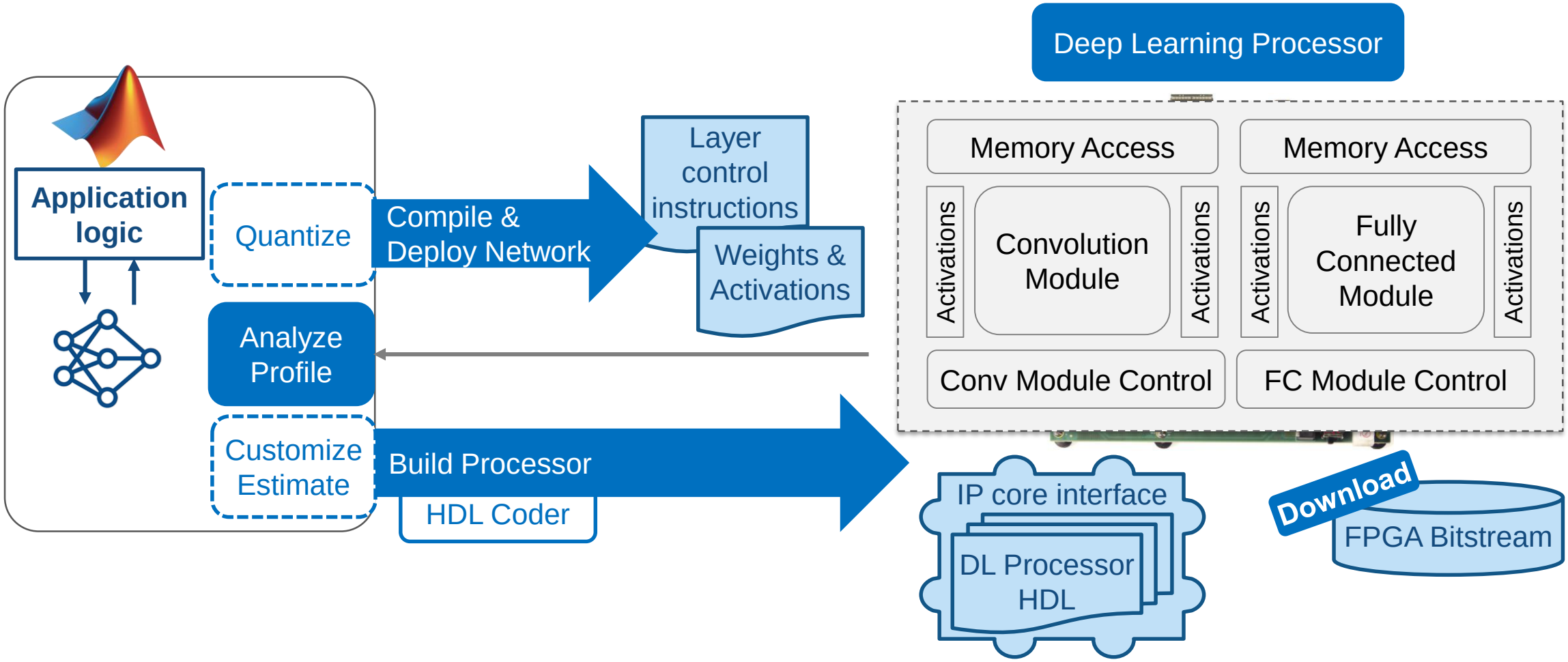


Deep Learning HDL Toolbox™

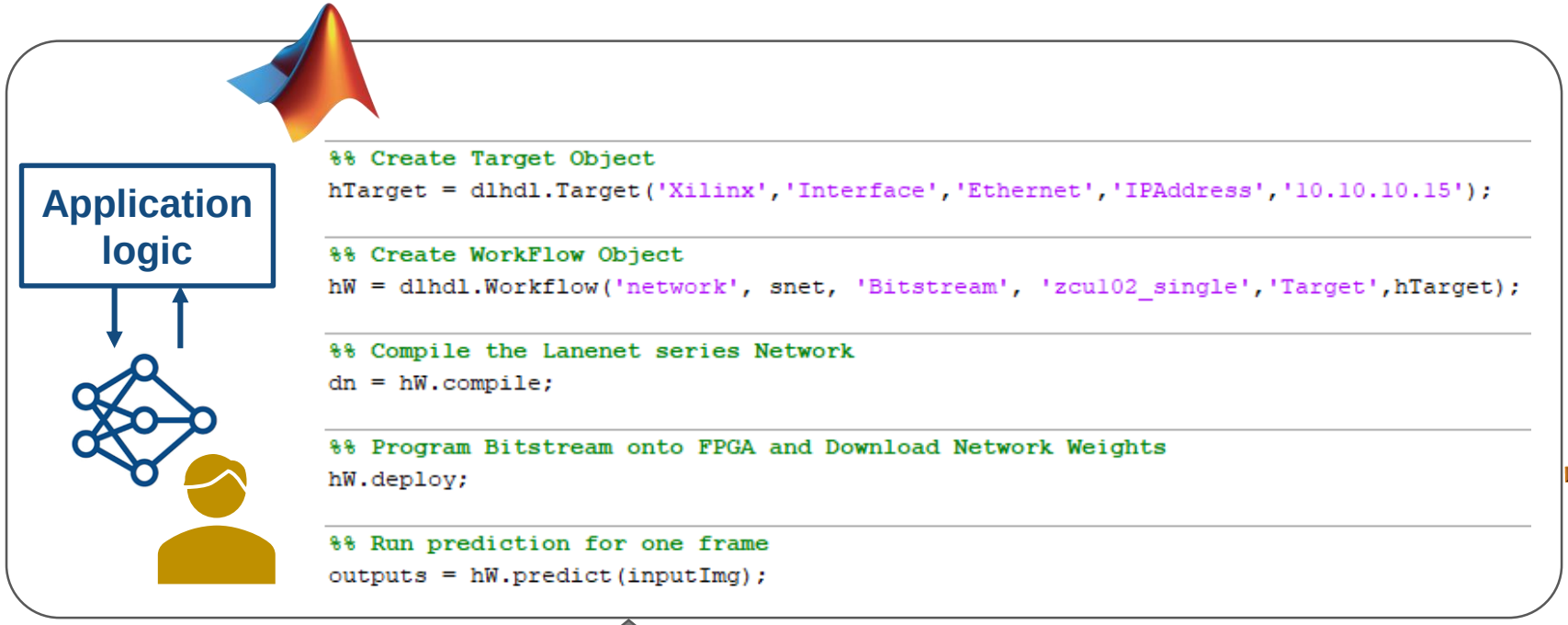
- Prototype network on FPGA
- Assess memory usage, latency, and accuracy
  - Adjust network and iterate
  - Quantize to fixed-point
- Generate customized deep learning processor HDL

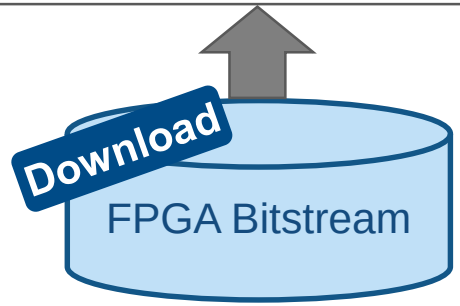
…all from within MATLAB!

Deep Learning HDL Toolbox Components



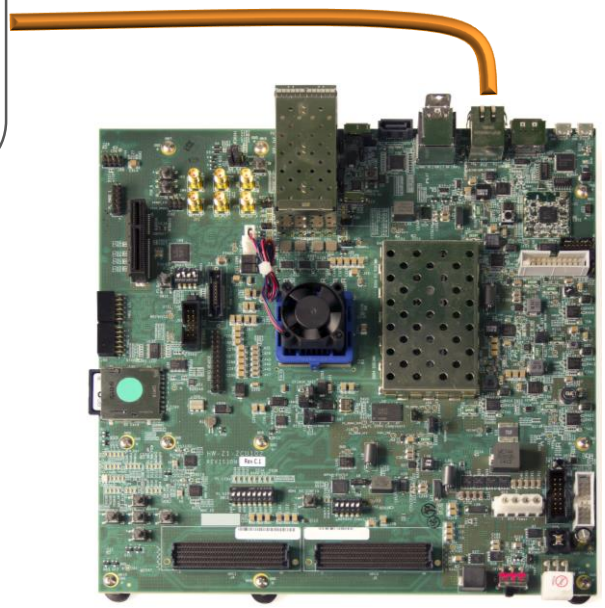
Get Started Prototyping on FPGA with Deep Learning HDL Toolbox™





Layer control instructions

Weights & Activations

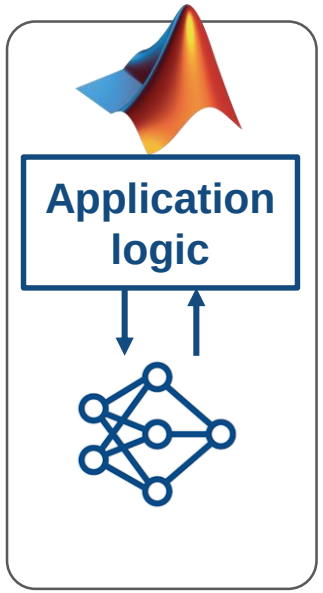


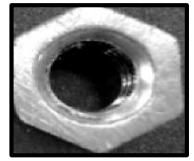
Hardware support package

Deep learning processor with I/O and external memory interfaces

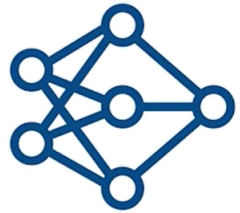
- Int8 or single
- Supported boards:
  - Xilinx: ZCU102 or ZC706
  - Intel: Arria10 SoC
- http://mathworks.com/hardware-support.html

Defect Detection Example



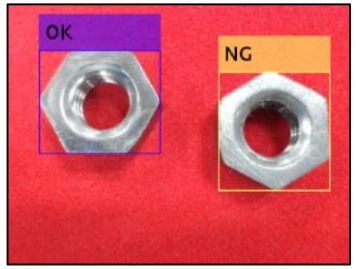


Pre-processing: Extract regions and resize



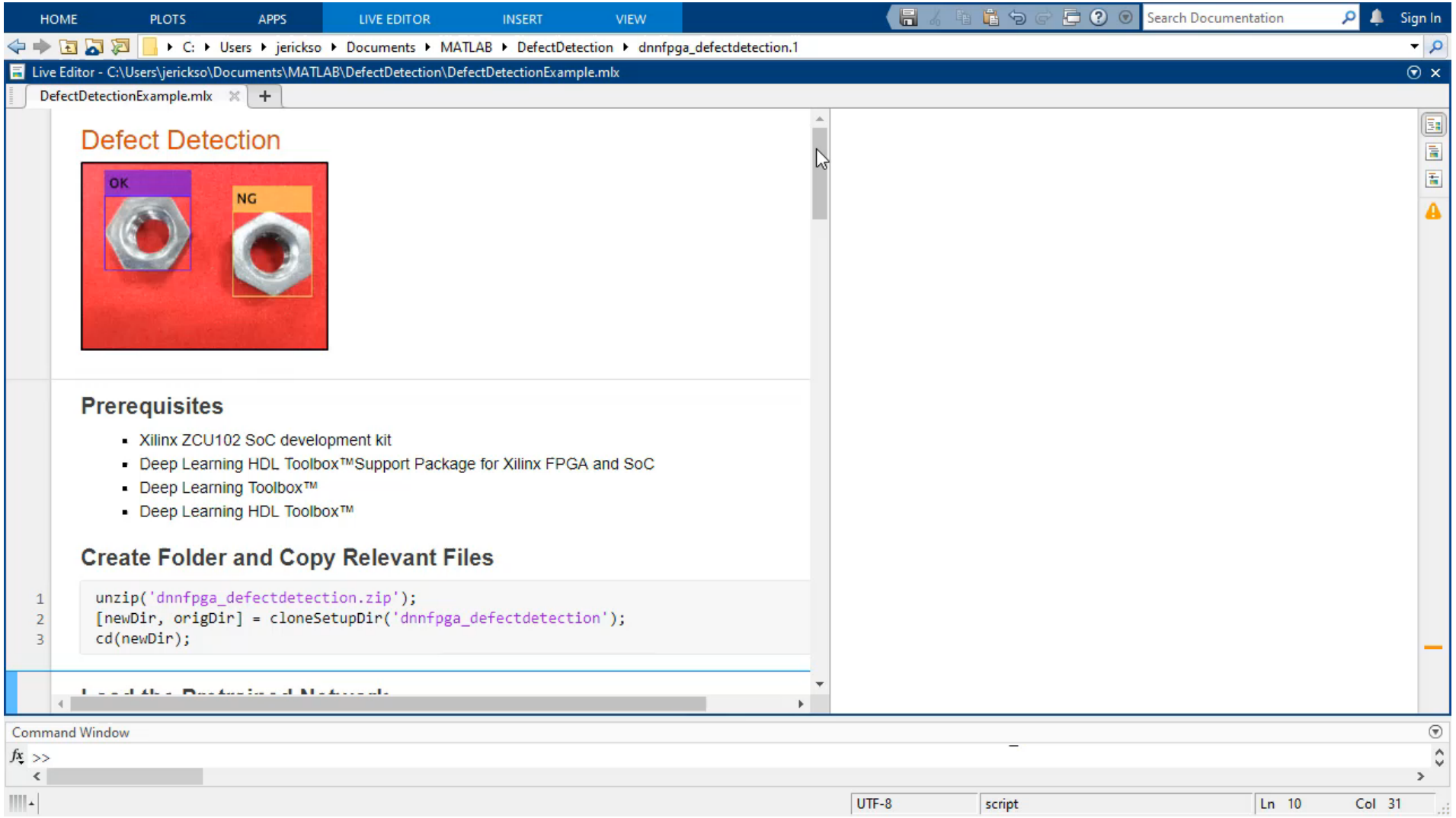
Inference: Predict using trained network

FPGA

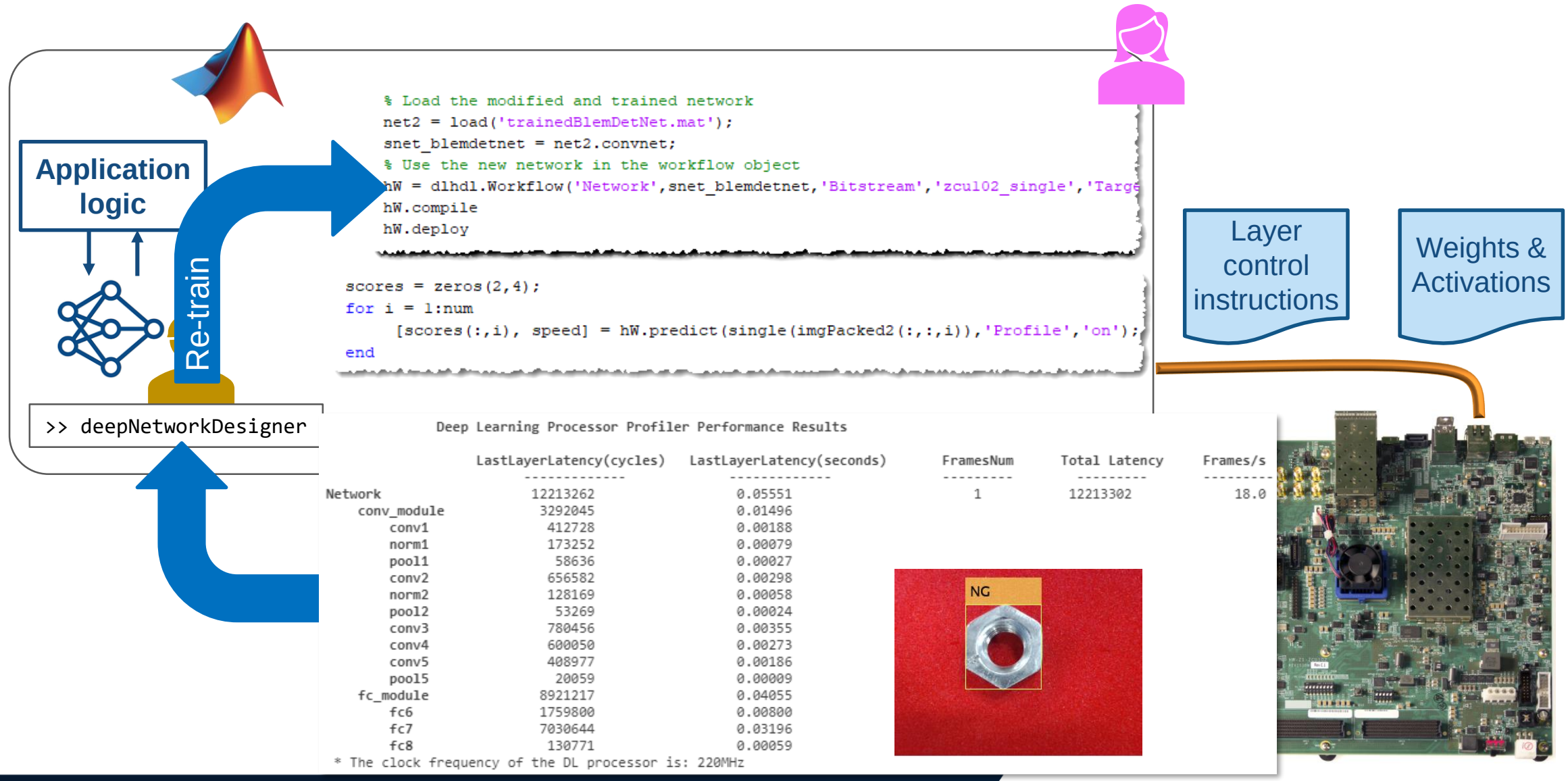


Post-processing: Annotate and label

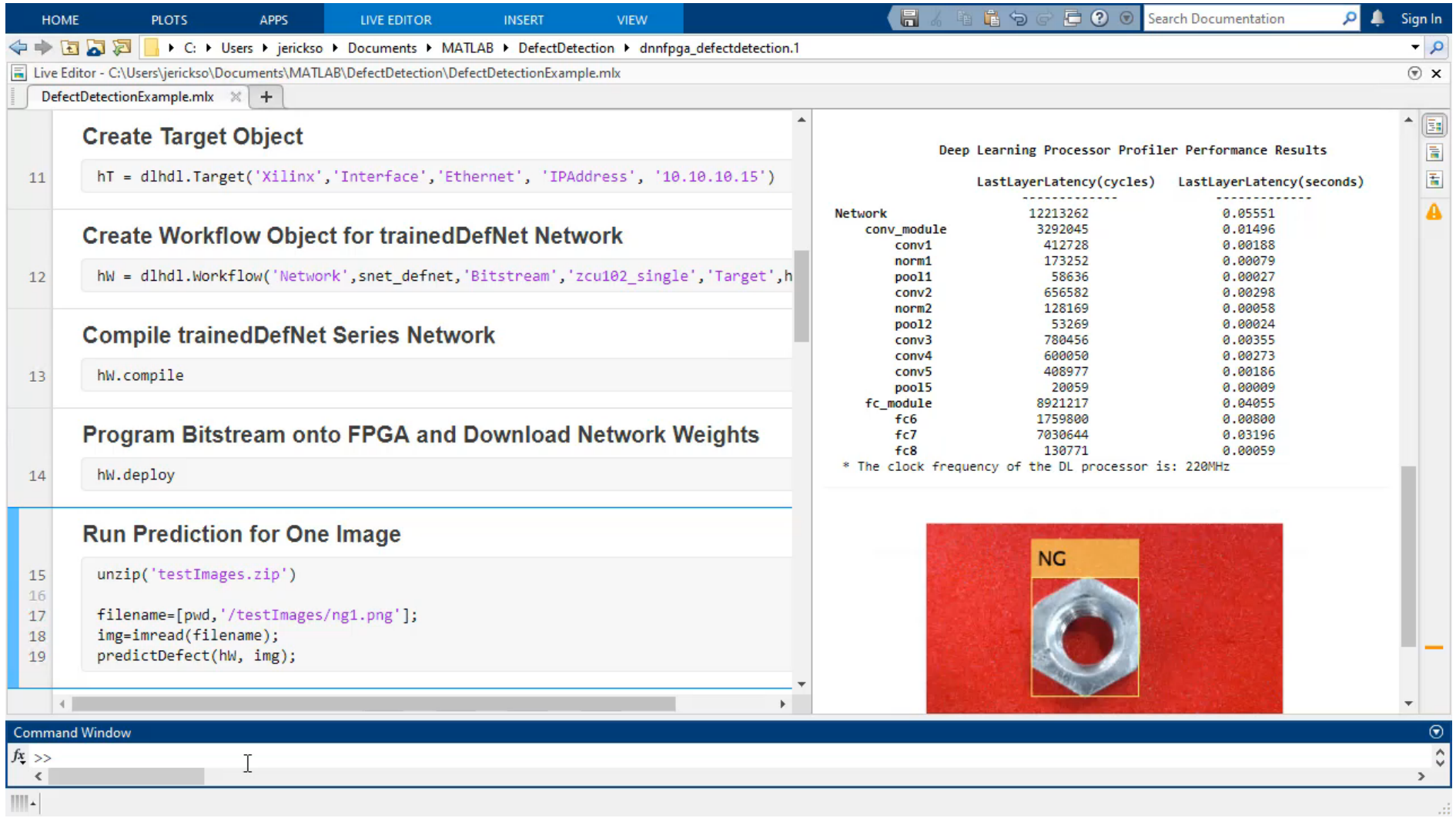
Run Deep Learning on FPGA from MATLAB



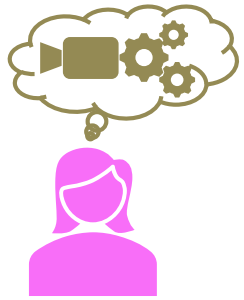
Profile FPGA Prototype and Iterate in MATLAB



Design Exploration and Customization



Collaborate to Quantize Network



Systems Engineer

Accuracy

Latency

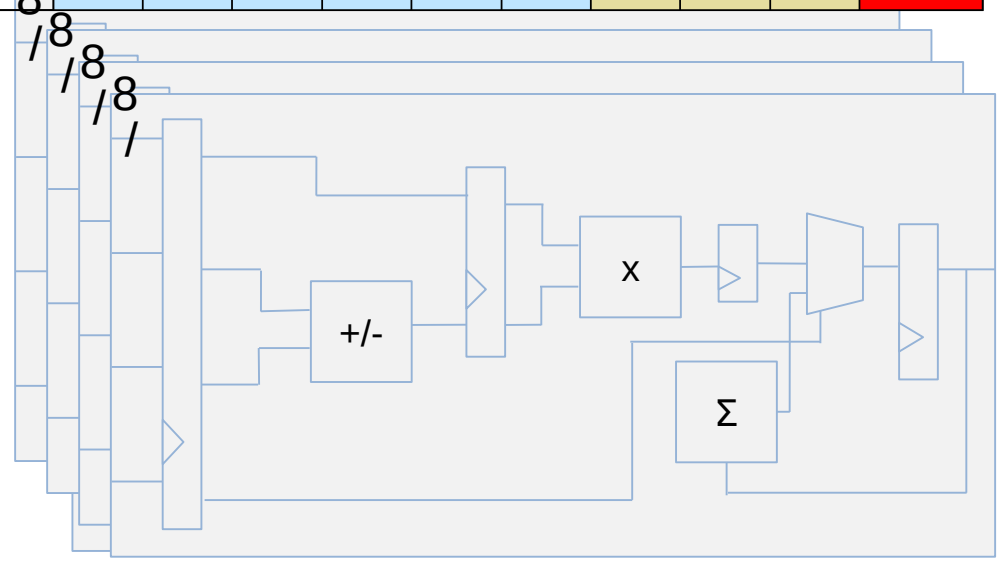
Cost

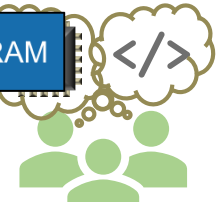
Power

Deep Learning Practitioner

	input	conv 1	conv 2	conv 3	conv 4	conv 5	fc6	fc7	fc8	Total
Parameters (Bytes)	n/a	35K	0.4M	0.9M	1.3M	0.5M	37M	16M	4M	58 M

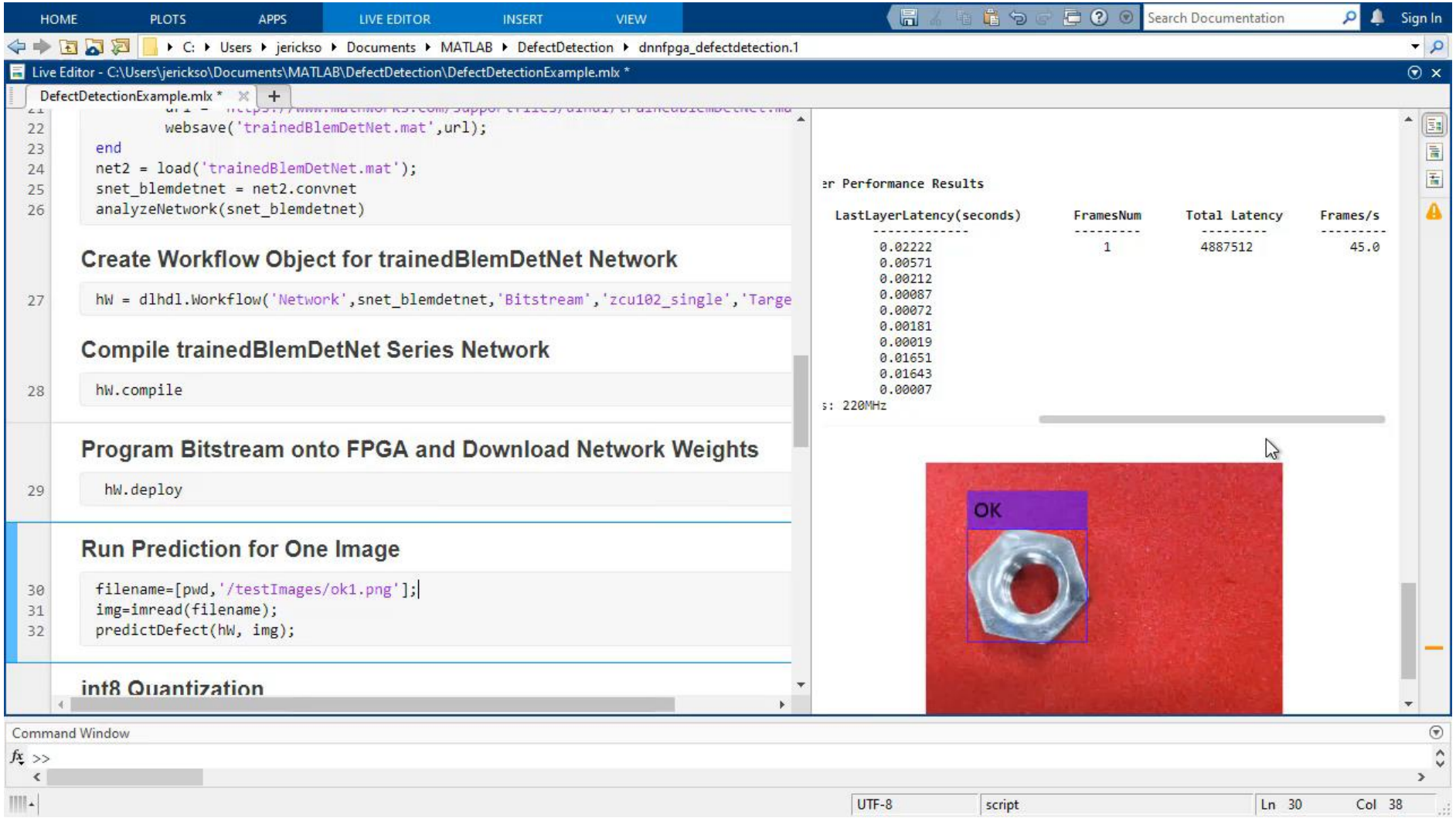
Off-chip RAM



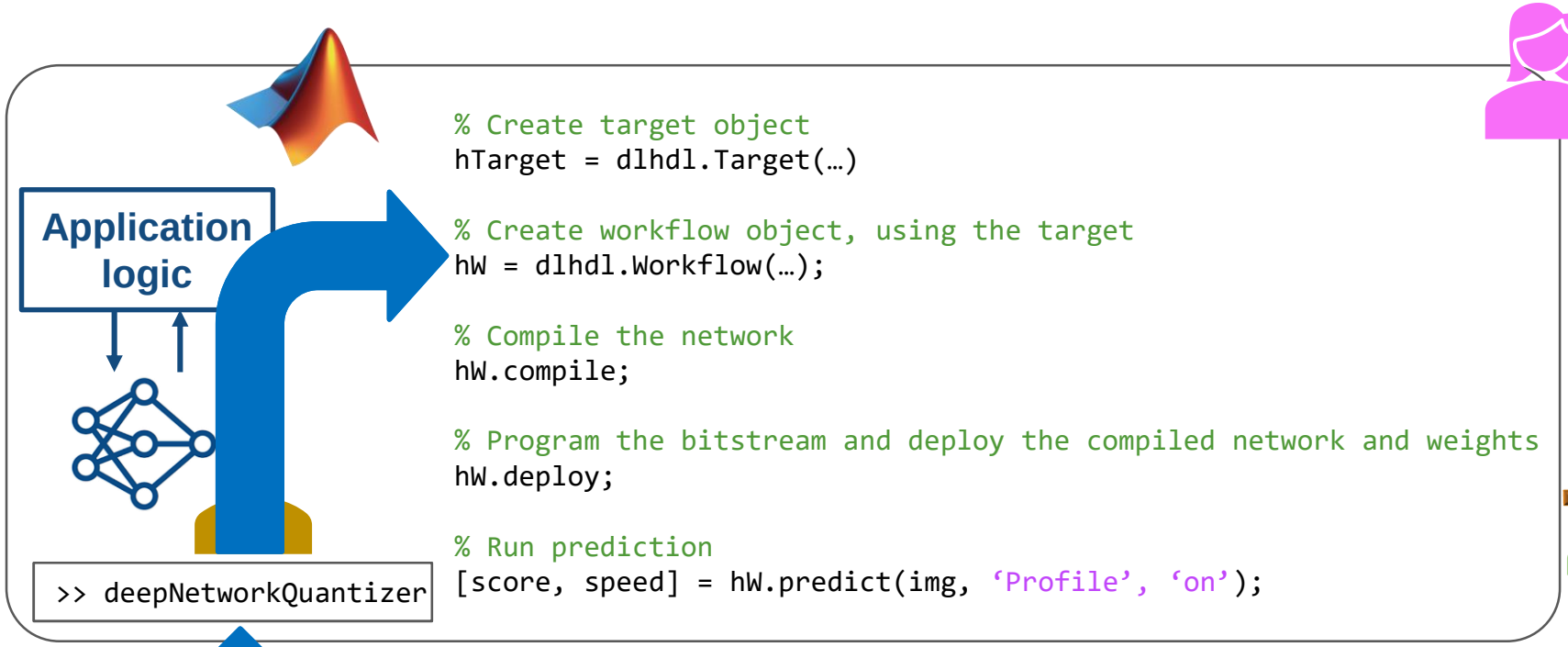


Hardware/Software Engineers

Int8 Quantization



Converge on an FPGA-Optimized Deep Learning Network



Quantize

Parameters	Speed
140 MB	18 fps
84 MB	45 fps
68 MB	139 fps

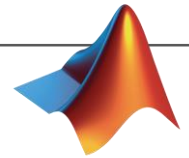
Generate HDL

Layer control instructions

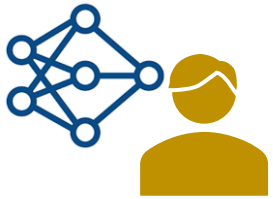
Weights & Activations

int8 Bitstream

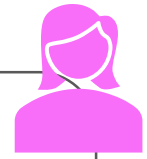
Generate Custom Deep Learning Processor HDL and IP Core



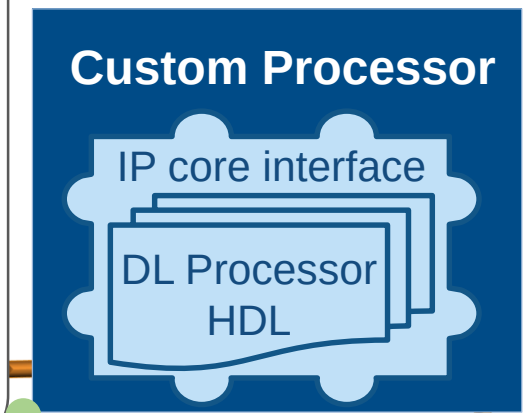
Application logic



```
% Create a custom processor object
hPC = dlhdl.ProcessorConfig;

% Customize processor characteristics
hPC.setModuleProperty('conv', 'KernelDataType',    'int8');
hPC.setModuleProperty('conv', 'ConvThreadNumber', 64);
hPC.setModuleProperty('fc', 'KernelDataType',    'int8');
hPC.setModuleProperty('fc', 'FCThreadNumber',    16);
hPC.TargetFrequency = 300;

% Create workflow object for this config, estimate performance
hW = dlhdl.Workflow('Network',quantizer,'ProcessorConfig',hPC)
hW.estimate('Performance');

% Generate HDL and IP core using HDL Coder
dlhdl.buildProcessor(hPC);
```

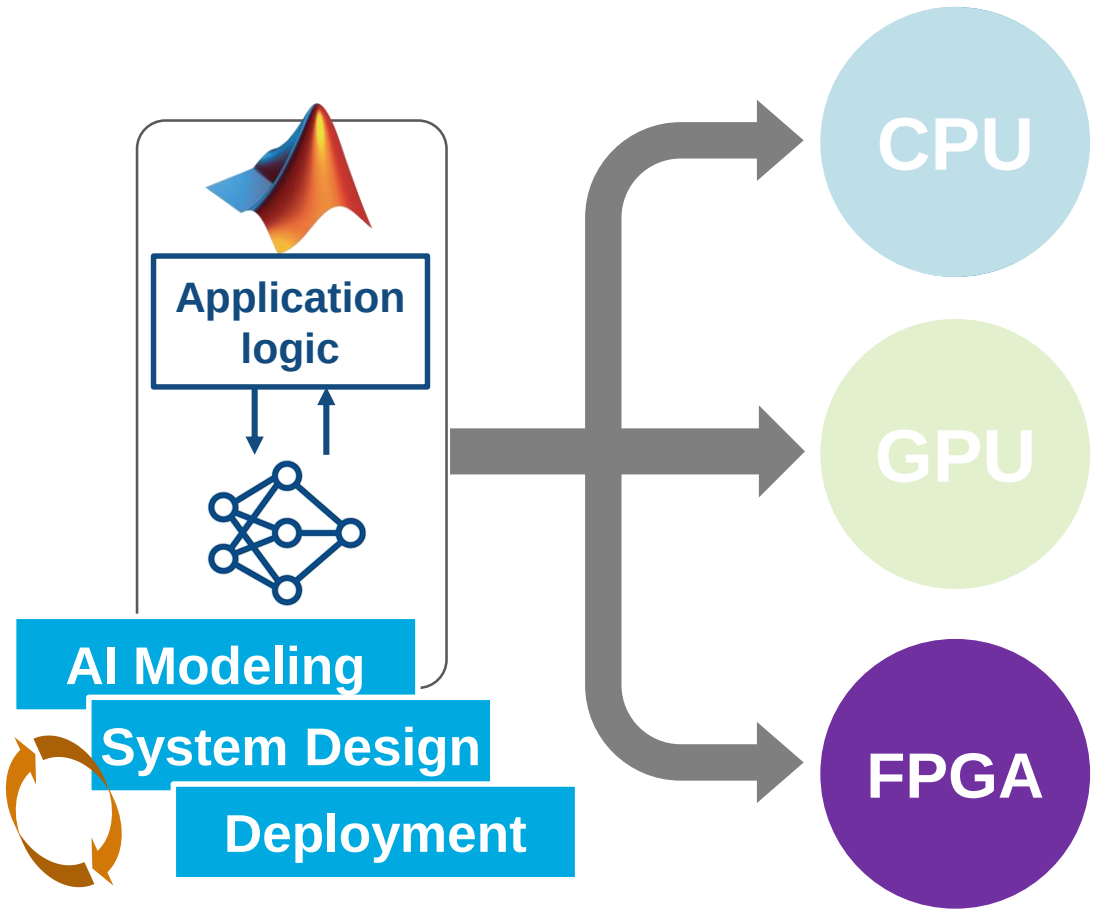


HDL Coder



- Configure processor settings
  - Parallel threads, frequency, memory sizes
  - Quantized or single precision floating point
  - Target frequency
- Target any hardware
  - Synthesizable RTL with AXI mappings
  - Automatic Xilinx or Intel implementation

Collaborate to Converge on Deep Learning FPGA Implementation



Deep Learning HDL Toolbox

- Prototype from MATLAB
- Tune for system requirements
- Configure and generate RTL

Learn More

- Deep Learning Solutions in MATLAB
  https://www.mathworks.com/solutions/deep-learning.html
- Deep Learning HDL Toolbox
  https://www.mathworks.com/products/deep-learning-hdl.html
- Onramp: Deep Learning in MATLAB
  https://www.mathworks.com/learn/tutorials/deep-learning-onramp.html
- MathWorks FPGA Solutions Page
  https://www.mathworks.com/solutions/fpga-asic-soc-development.html