Deep Learning and Computer Vision on Zynq using reVISION/SDSoC
## Applications: Wide Range of Rapidly Changing Vision Guided Systems

<table>
<thead>
<tr>
<th>Embedded Vision Systems</th>
<th>Vision Guided Autonomous Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factory Robotics</td>
<td>Vision Guided ‘Cobots’</td>
</tr>
<tr>
<td>Camera Equipped Aircraft</td>
<td>‘Sense and Avoid’ &amp; Autonomous Drones</td>
</tr>
<tr>
<td>Physical Displays and HMI</td>
<td>Augmented Reality and HUDs</td>
</tr>
<tr>
<td>Forward Auto Camera</td>
<td>Autonomous Vehicles</td>
</tr>
<tr>
<td>Video Security Cams</td>
<td>Automated Surveillance</td>
</tr>
<tr>
<td>Medical Imaging and Human Eye</td>
<td>Automated Medical Diagnostics</td>
</tr>
</tbody>
</table>

**Target Markets:** Prosumer - Automotive - Industrial - Medical - A&D
Application Mandates: From Embedded Vision to Autonomous Systems

Intelligent and Immediate Response with Efficiency

Flexibility to Upgrade to Latest Algorithms & Sensors

Always Connected to Other Machines and the Cloud
Xilinx Unique Application Advantages

Responsive

Optimized from Sensors to <8-bit Inference & Control

Reconfigurable

Reconfigurable for Latest Networks & Sensors

Connected

Any-to-Any Connectivity
OpenVX™ Vision Acceleration

Caffe

Application Development

OpenCV

DNN
CNN
GoogLeNet
SSD
FCN …

Algorithm Development

SDSoC™ Environment

Platform Development
## Zynq SoCs Offer Superior Throughput, Latency and Efficiency

### Machine Learning Inference

<table>
<thead>
<tr>
<th></th>
<th>Xilinx ZU9</th>
<th>Xilinx ZU5</th>
<th>Nvidia TX1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images/s</td>
<td>370.0</td>
<td>155.0</td>
<td>70</td>
</tr>
<tr>
<td>Power (W)</td>
<td>7.0</td>
<td>4.5</td>
<td>7.9</td>
</tr>
<tr>
<td>Images/s/watt</td>
<td>53.0</td>
<td>34.5</td>
<td>8.9</td>
</tr>
</tbody>
</table>

**Images/sec/watt**: **6x**

**Latency (ms)**: **1/5**

### Real Time Applications Latency

<table>
<thead>
<tr>
<th></th>
<th>Xilinx ZU9</th>
<th>Xilinx ZU5</th>
<th>Nvidia TX1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images/s</td>
<td>370.0</td>
<td>155.0</td>
<td>70</td>
</tr>
<tr>
<td>Latency (ms)</td>
<td>2.7</td>
<td>6.4</td>
<td>14.2</td>
</tr>
</tbody>
</table>

**Images/s**: **370.0**

**Latency (ms)**: **2.7**

### GoogLeNet @ batch = 8

<table>
<thead>
<tr>
<th></th>
<th>Xilinx ZU9</th>
<th>Xilinx ZU5</th>
<th>Nvidia TX1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Images/s</td>
<td>370.0</td>
<td>155.0</td>
<td>163</td>
</tr>
<tr>
<td>Latency (ms)</td>
<td>2.7</td>
<td>6.4</td>
<td>49.0</td>
</tr>
</tbody>
</table>

**Images/s**: **370.0**

**Latency (ms)**: **2.7**

---

For large batch, CPU/GPU/DSPs latency increases significantly.

---

- ML based on Xilinx GoogleNet performance roadmap
- All benchmarks utilize as much resources as possible on GPU (~99%) and programmable logic (~70%)
The Divergence of Training and Inference in Machine Learning

**Training**: Process for machine to “learn” and optimize model from data

**Inference**: Using trained models to predict/estimate outcomes from new observations in efficient deployments

**Top-5 Accuracy**

<table>
<thead>
<tr>
<th>Model</th>
<th>FP-32</th>
<th>FIXED-16 (INT16)</th>
<th>FIXED-8 (INT8)</th>
<th>Difference vs FP32</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td>86.6%</td>
<td>86.6%</td>
<td>86.4%</td>
<td>(0.2%)</td>
</tr>
<tr>
<td>GoogLeNet</td>
<td>88.6%</td>
<td>88.5%</td>
<td>85.7%</td>
<td>(2.9%)</td>
</tr>
<tr>
<td>SqueezeNet</td>
<td>81.4%</td>
<td>81.4%</td>
<td>80.3%</td>
<td>(1.1%)</td>
</tr>
</tbody>
</table>

*Inference now 8 bit and below for maximum efficiency*

Inference Precisions Moving to Lower and Variable Precision

Citation: https://arxiv.org/pdf/1510.00149.pdf
BNN: Unparalleled Performance

- Reducing precision from 8b to 1b shrinks LUT cost by 40x
- Potential to scale CNN performance to above 23TOPS (ZU9)

<table>
<thead>
<tr>
<th>1b</th>
<th>1b</th>
<th>1b</th>
<th>1b</th>
<th>1b</th>
<th>1b</th>
<th>1b</th>
<th>1b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
<tr>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
<td>1b</td>
</tr>
</tbody>
</table>

- Assuming 300 MHz with 90%/70% DSP/LUT utilizations
- Resource consumption assumption: 2.5 LUTs/op (INT1), 16 LUTs/op (INT4), 0.25 DSP/op (INT8)
BNN: Unparalleled Performance

- Reducing precision from 8b to 1b shrinks LUT cost by 40x
- Potential to scale CNN performance to above 23TOPS (ZU9)

• Assuming 300 MHz with 90%/70% DSP/LUT utilizations
• Resource consumption assumption: 2.5 LUTs/op (INT1), 16 LUTs/op (INT4), 0.25 DSP/op (INT8)
BNN: Unparalleled Performance

- Reducing precision from 8b to 1b shrinks LUT cost by 40x
- Potential to scale CNN performance to above 23TOPS (ZU9)

- Assuming 300 MHz with 90%/70% DSP/LUT utilizations
- Resource consumption assumption: 2.5 LUTs/op (INT1), 16 LUTs/op (INT4), 0.25 DSP/op (INT8)
- 10W power assumption on ZU9

© Copyright 2017 Xilinx
Small expense in accuracy but fast improving
– Agreement less precision is sufficient
Future Proof Architecture for Any Precisions

- **CPU**
  - Limited to 32 bit operations
  - SIMD operations at INT8/16
- **GPU**
  - New devices required to support change in precision efficiently
- **Xilinx**
  - Reconfigurable to scale and optimize for different precisions
Low Latency Inference by Layer to Layer Dataflow On Chip

On-chip Memory

<table>
<thead>
<tr>
<th>Nvidia Tegra X1 (GPU Regfile + L2)</th>
<th>Xilinx ZU7 (BRAM + URAM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Mb</td>
<td>38 Mb</td>
</tr>
</tbody>
</table>

Up to 6x More On-chip Memory than SoCs and eGPUs

xFdnn: Direct Deep Learning Inference from Caffe

1. Import .prototxt and trained weights

2. Call prototxt runtime API in your application

3. Cross-compile for Cortex-A53 and run on a board

Compiles only ARM software code in minutes. No hardware compilation.
Deep Learning Design Examples

<table>
<thead>
<tr>
<th>Model</th>
<th>Batch</th>
<th>Gops/img</th>
<th>Images/s</th>
<th>Power (W)</th>
<th>Images/s/watt</th>
<th>May 2017</th>
<th>Roadmap</th>
</tr>
</thead>
<tbody>
<tr>
<td>GoogLeNet</td>
<td>1</td>
<td>3.2</td>
<td>121</td>
<td>6.0</td>
<td>20.2</td>
<td></td>
<td>370</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7.0</td>
</tr>
<tr>
<td>SSD</td>
<td>1</td>
<td>62.4</td>
<td>6.3</td>
<td>6.0</td>
<td>1.1</td>
<td></td>
<td>52.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCN-AlexNet</td>
<td>1</td>
<td>42.0</td>
<td>7.0</td>
<td>6.0</td>
<td>1.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VGG-16</td>
<td>1</td>
<td>30.9</td>
<td>14.5</td>
<td>6.0</td>
<td>2.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AlexNet</td>
<td>1</td>
<td>1.4</td>
<td>92</td>
<td>6.0</td>
<td>15.3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Programmable Logic running at 300 MHz, Input size: GoogLeNet, AlexNet, VGG-16 = 224x224, SSD = 300x300, FCN=480x480

© Copyright 2017 Xilinx
Performance & Area Scalability

Resource Util w/ DSP Scaling

- BRAM (in tens)
- LUT (in thousands)
- FF (in thousands)

Performance Scaling w/ DSPs (GoogleNet)

#DSPs
- 128
- 256
- 448
- 704
- 1152
- 1344
- 1664

Perf (images/sec)
- 0
- 20
- 40
- 60
- 80
- 100
- 120

Perf (fps)
- 0
- 20
- 40
- 60
- 80
- 100
- 120

Roadmap to 370 img/s
Caffe Prototxt to hardware operators
DeepX: Deep Learning Inference Processor

- Parameterized design.
- Scalable
- Rich Instruction Set with 30+ opcodes
- Mixed Precision support (16b, 8b)
Deep Learning IP Export Flow (Roadmap)

- Export DNN IP and ARM scheduler to integrate into real system
- Compile-time configuration of DNN IP (e.g. DSP, BRAM, buffer size …)

```c
main()
{
  imread(A);
  imread(B);
  roi_crop(A,img)
  xFdnn <DSP,BRAM,BUF,...>(img,prototxt,weights,out)
  imshow(out);
}
```
Putting It All Together: CV and CNN with Multiple Sensors

Dual 1280x720 @ 30 FPS

1280x720 @ 60 FPS

Available as Black-box Demo in 2017.1 and Design Example in 2017.3
ZCU102 (ZU9)

ARM Cortex-A53

SDSoC Application

Video Lib

App Stub

Video Lib

V4L2

DM* Driver

DRM

Linux

Need to select one source

Need to select one sink

USB3

DDR

HDR

ISP/VPSS*

Optical Flow

CNN

HDR

DDR

DDR

HDMI

DP

HDMI

Stereoscopic Depth Map

SDSoC Generated

SDSoC Platform

Embedded Vision Platform 2017.1
## Embedded Vision Development Kits

<table>
<thead>
<tr>
<th>Base Zynq Board</th>
<th>ZCU102</th>
<th>ZC702</th>
<th>ZC706</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>ZU9 (16nm)</td>
<td>Z7020 (28nm)</td>
<td>Z7045 (28nm)</td>
</tr>
<tr>
<td>CPU</td>
<td>Quad Cortex A53 up to 1.5GHz</td>
<td>Dual Cortex A9 up to 1.0GHz</td>
<td></td>
</tr>
<tr>
<td>Peak GOPS @ INT8</td>
<td>7857</td>
<td>571</td>
<td>2331</td>
</tr>
<tr>
<td>On-chip RAM (Mbits)</td>
<td>32.1</td>
<td>4.9</td>
<td>19.1</td>
</tr>
<tr>
<td>Inputs</td>
<td>USB3, MIPI, HDMI</td>
<td>HDMI*</td>
<td>HDMI*</td>
</tr>
<tr>
<td>Outputs</td>
<td>HDMI, DisplayPort</td>
<td>HDMI</td>
<td>HDMI</td>
</tr>
<tr>
<td>Video Codec Units</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>reVISION Support</td>
<td>xFopencv, xFdnn</td>
<td>xFopencv, xFdnn</td>
<td>xFopencv, xFdnn</td>
</tr>
</tbody>
</table>

### Sensor Inputs

<table>
<thead>
<tr>
<th>Spec</th>
<th>Interface</th>
<th>Spec</th>
<th>Interface</th>
<th>Spec</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>3840x2160 @ 60 FPS</td>
<td>MIPI via FMC</td>
<td>3840x1080 @ 30 FPS</td>
<td>USB3</td>
<td>1920x1080 @ 60 FPS</td>
<td>USB3</td>
</tr>
</tbody>
</table>

* Requires an HDMI IO FMC card