FPGA加速机器学习应用

罗霖
Andy.luo@Xilinx.com

2017年6月20日
Xilinx – The All Programmable Company

$2.21B FY16 revenue
>57% market segment share
3,500+ employees worldwide

20,000 customers worldwide
3,500+ patents
60 industry firsts

XILINX - Founded 1984

Headquarters
Research and Development
Sales and Support
Manufacturing
Great Partnership between Mathworks & Xilinx

Xilinx Zynq Support from Computer Vision System Toolbox

Design and prototype vision systems using Xilinx Zynq-based hardware
What is FPGA

- A field-programmable gate array (FPGA) is an integrated circuit that can be programmed in the field after manufacture.

- FPGA contain an array of programmable logic blocks and a hierarchy of reconfigurable interconnects that allow the blocks to be "wired together".

- Usually programmed with HDL (VHDL/Verilog) and now supports C/C++/OpenCL and model-based tool (Matlab, Labview…)

- A very wide range of applications including wired&wireless communication, data center, aerospace&defense, industrial, medical, automotive, test&measurement, audio&video, even consumer…
CPU vs GPU vs FPGA

- Complex control logic
- Large caches
- Optimized for serial operations

- Limited control function
- High throughput
- Built for parallel operations

- Many programmable I/O
- Large internal memory
- Customized for complex control & parallel computation
SOC with FPGA

ARM Subsystem

High-speed on-chip bus

28nm SoC

FPGA Subsystem

16nm SoC

28nm SoC

FPGA Subsystem

16nm SoC
Autonomous Vehicles
Medical Bioinformatics
Industrial IOT
Surveillance
Financial
Ecommerce
Social
Cloud Acceleration
Security
Ecommerce Social
Autonomous Vehicles
Industrial IOT
Medical Bioinformatics
Surveillance
**Training**: Process for machine to “learn” and optimize a model from data

**Inference**: Using trained model to predict/estimate outcomes from new observations

---

Training:

- Large $N$
- Forward
- Backward
- Error
- "dog"?
- "human face"

Inference:

- Smaller, varied $N$
- Forward
- "human face"

---

**Deployment**

- R&D server cluster
- HPC Servers (GPU-enabled)
- Cloud servers (FPGAs, ASIC, GPU)
- Edge Devices (FPGAs, ASIC, GPU)

---

**Xilinx Focus**
Deep Learning Technical Challenges

- Computational Intensive
- Memory Bandwidth Intensive
- Deployment Power Efficiency

Example – Deep Learning Inference: Image Classification (AlexNet)

Compute Operations: 2,270,000,000
Data Movements: 65,000,000

"Dog"
Deep Compression

- Small DNN models are critical.

<table>
<thead>
<tr>
<th>Network</th>
<th>Original Size</th>
<th>Compressed Size</th>
<th>Compression Ratio</th>
<th>Original Accuracy</th>
<th>Compressed Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>240MB</td>
<td>6.9MB</td>
<td>35x</td>
<td>80.27%</td>
<td>80.30%</td>
</tr>
<tr>
<td>VGGNet</td>
<td>550MB</td>
<td>11.3MB</td>
<td>49x</td>
<td>88.68%</td>
<td>89.09%</td>
</tr>
<tr>
<td>GoogleNet</td>
<td>28MB</td>
<td>2.8MB</td>
<td>10x</td>
<td>88.90%</td>
<td>88.92%</td>
</tr>
<tr>
<td>SqueezeNet</td>
<td>4.8MB</td>
<td>0.47MB</td>
<td>10x</td>
<td>80.32%</td>
<td>80.35%</td>
</tr>
</tbody>
</table>

30x – 50x compression rate without hurting accuracy
Machine Learning Moving towards Lower Precision
Activation Quantization: 8 Bits Are Enough

- Inference with Integer Quantization
  - Fixed-Point sufficient for Deployment (INT16, INT8)
  - No Significant Loss in Accuracy (< 1%)
  - >10x Energy Efficiency OPs/J (INT8 vs FP32)
  - 4x Memory Energy Efficiency Tx/J (INT8 vs FP32)

<table>
<thead>
<tr>
<th>Model</th>
<th>FP32</th>
<th>FIXED-16</th>
<th>FIXED-8</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top-1</td>
<td>65.77%</td>
<td>65.78%</td>
<td>65.58%</td>
</tr>
<tr>
<td>Top-5</td>
<td>86.64%</td>
<td>86.65%</td>
<td>86.38%</td>
</tr>
<tr>
<td>GoogLeNet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top-1</td>
<td>68.60%</td>
<td>68.70%</td>
<td>62.75%</td>
</tr>
<tr>
<td>Top-5</td>
<td>88.65%</td>
<td>88.45%</td>
<td>85.70%</td>
</tr>
<tr>
<td>SqueezeNet</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top-1</td>
<td>58.69%</td>
<td>58.69%</td>
<td>57.27%</td>
</tr>
<tr>
<td>Top-5</td>
<td>81.37%</td>
<td>81.35%</td>
<td>80.32%</td>
</tr>
</tbody>
</table>
FPGA Advantages in Deep Learning

Customizable Massive Parallel Compute Power

Fine-grained Memory Hierarchy Reduce Memory Bottlenecks

Power Efficient
Xilinx supports up to 27x18 bits in a single multiplier vs. 18x18 in Arria/Stratix 10 DSP Block

Enough bit-width to perform two separate MACCs with one shared factors for 8-bit computes on single DSP
ML Performance Comparison with Nvidia Tegra Devices

Nvidia Tegra K1/X1 SoC
- 28 nm / 20nm
- 192 / 256 CUDA Cores
- Caffe with latest CuDNN

Xilinx Zynq 7020/ZU2CG (Projection)
- 28nm / 16nm
- 85k/103k logic cells
- 220/240 DSP
- 4.9/5.3Mb BRAM

Benchmark

VGG16
Image classification
30.68 Gop, 13 Conv layers

YOLO Tiny
General object detection
5.54 Gop, 9 Conv layers

Customized Network
Face alignment
104.6 Mop, 9 Conv layers
ML Performance Comparison with Nvidia Tegra Devices (Cont.)

Source: Deephi
Zynq7020 PL @ 200MHz, ZU2CG PL @ 300MHz
Different User Personas

Hardware Engineer

Algorithm / DSP Engineer

Software Engineer
MathWorks Guided Workflow for Zynq

- From requirements, to model, to rapid prototype
- A guided workflow for hardware and software development
  - HDL Coder: Programmable Logic bitstream generation
  - Embedded Coder: Software build file generation / Drivers
Accelerate Deep Learning Prototype on Zynq with Matlab/Simulink

Algorithm Development

Neural Network Toolbox

MatConvNet

http://www.vlfeat.org/matconvnet/

Automated Code Generation

Target HW Deployment
Separate platform design from differentiated logic
  - Let application designers focus on the differentiated logic

Spend less time on the standard connectivity
  - **IPI**: configure & generate a platform on a custom board
  - Use of Partial Reconfiguration to guarantee performance

Spend more time on the differentiated logic
  - **HLS**: enabling core technology: C/C++/OpenCL synthesis
    - Exhaustive simulation, architecture exploration, code portability
  - **HLx**: Accelerates HW design: IP design (HLS/SysGen) + connectivity platform integration (IP Integrator)
  - **SDx**: Brings SW programmability to FPGA based platform
The SDSoC Development Environment

- ASSP-like programming experience
- System-level profiling
- Full system optimizing compiler
- Expert use model for platform developers and system architects
## reVISION Stack

<table>
<thead>
<tr>
<th>OpenVX</th>
<th>Caffe</th>
<th>Application Development</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenCV</td>
<td>DNN</td>
<td>Algorithm Development</td>
</tr>
<tr>
<td>SDSs</td>
<td>CNN</td>
<td></td>
</tr>
</tbody>
</table>

| GoogleNet | BID | FCN |

**MACHINE LEARNING | COMPUTER VISION | SENSOR FUSION | CONNECTIVITY**
Compiles only ARM software code in minutes. No hardware compilation.
DeepX: Deep Learning Inference Processor

- Parameterized design. Scalable.
- Rich Instruction Set with 30+ opcodes.
  - Support for all popular networks.
- Mixed Precision support (16b, 8b).
- Simple Usage Model.
## CNN Functions in different networks

<table>
<thead>
<tr>
<th>Function/CNN</th>
<th>AlexNet</th>
<th>AlexNet FCN</th>
<th>VGG</th>
<th>GoogleNet</th>
<th>SqueezeNet</th>
<th>PVANet</th>
<th>ResNet</th>
<th>SSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolution (2D)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>ReLU activation</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>CReLU</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Fully connected</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>SoftMax</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Deconv</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Dilation</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>NMS</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Permute</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Maxpool</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Avg Pool</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Concat</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Eltwise</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>LRN Norm</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>L2 Norm</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Batch Norm</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>
### Deep Learning Design Examples

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GoogLeNet</td>
<td>1</td>
<td>121</td>
<td>6.0</td>
<td>20.2</td>
<td>7.0</td>
<td>52.9</td>
</tr>
<tr>
<td>SSD</td>
<td>1</td>
<td>6.3</td>
<td>6.0</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FCN-AlexNet</td>
<td>1</td>
<td>7.0</td>
<td>6.0</td>
<td>1.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VGG-16</td>
<td>1</td>
<td>14.5</td>
<td>6.0</td>
<td>2.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AlexNet</td>
<td>1</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
## Development Kits

<table>
<thead>
<tr>
<th>Base Zynq Board</th>
<th>ZCU102</th>
<th>ZCU104</th>
<th>ZC702</th>
<th>ZC706</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>ZU9 (16nm)</td>
<td>ZU7 (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>
<td></td>
</tr>
<tr>
<td>Peak GOPS @ INT8</td>
<td>7857</td>
<td>5386</td>
<td>571</td>
<td>2331</td>
</tr>
<tr>
<td>On-chip RAM (Mbits)</td>
<td>32.1</td>
<td>38.0</td>
<td>4.9</td>
<td>19.1</td>
</tr>
<tr>
<td>Inputs</td>
<td>USB3, MIPI, HDMI</td>
<td>USB3, MIPI, HDMI</td>
<td>HDMI*</td>
<td>HDMI*</td>
</tr>
<tr>
<td>Outputs</td>
<td>HDMI, DisplayPort</td>
<td>HDMI, DisplayPort</td>
<td>HDMI</td>
<td>HDMI</td>
</tr>
<tr>
<td>Video Codec Units</td>
<td>No</td>
<td>4K60 Encode/Decode</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>
<td>xFopencv, xFdnn</td>
</tr>
</tbody>
</table>

### Sensor Inputs

<table>
<thead>
<tr>
<th>Spec</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>MIPI via FMC</td>
</tr>
<tr>
<td>*</td>
<td>USB3</td>
</tr>
</tbody>
</table>

* Requires an HDMI IO FMC card
Optical Flow + Stereo Vision + Pedestrian Detection with Multiple Sensors

Dual 1280x720 @ 30 FPS

1280x720 @ 60 FPS
Summary

- Machine learning inference poses great challenges for embedded system in computation and memory bandwidth

- FPGA is very suitable for machine learning inference

- Model-based design and optimized libraries accelerate customer design for machine learning applications
Resources

- Deep Learning with INT8 Optimization on Xilinx Devices

- Reduce Power and Cost by Converting from Floating Point to Fixed Point

- Xilinx reVISION developer zone

- Xilinx Reconfigurable Acceleration Stack Accelerates Mainstream Adoption of Xilinx FPGAs in Hyperscale Data Centers

- WP477 UltraRAM: Breakthrough Embedded Memory Integration on UltraScale+ Devices

- Virtex UltraScale+ FPGAs with HBM Technology