# General Reading

## FPGA-based accelerator for convolution operations

* {b1}
* <https://ezproxyprod.ucs.louisiana.edu:2373/document/9172934>
* Maps convolution to matrix convolution for flexibility/continuity, mapping this way allows for the acceleration of the convolution to ignore system design constraints
* Proposes a Systolic Array of Processing Elements (PE) to accelerate CNN
* Each PE shifts data and completes MAC operation
* Low level approach to convolution

## Angel-Eye: A Complete Design Flow for Mapping CNN Onto Embedded FPGA

* {b2}
* <https://ezproxyprod.ucs.louisiana.edu:2373/document/7930521>
* Has previous work (Here is a presentation of it): <http://nicsefc.ee.tsinghua.edu.cn/media/publications/2016/FPGA2016_None_slide.pdf>
* Surveys current CNN implementations (as of 2017)
* Proposes a programmable and flexible CNN accelerator architecture
* States that many CNN architectures are too bulky for embedded systems/IoT
* Describes CNN Layers
  + Convolution layer
    - Applies a trained filter value to an input feature map to extract local features. Usually cascade several layers to extract many features.
  + Fully Connected layer
    - Usually, a classifier stage
  + Nonlinearity layer
    - Help with fitting, usually ReLU
  + Pooling layer
    - Down sampling, usually average or max
* States modern CNN prefer smaller (3x3/5x5) kernel size
* Chooses to compress the CNN model
  + States that an effective method is to limit bit width to ~16b or 12b
* This paper is not exactly talking about reconfigurable hardware per say but a design flow to reconfigure a CNN model for different hardware applications

## A FPGA-based Accelerator of Convolutional Neural Network for Face Feature Extraction

* {b3}
* <https://ezproxyprod.ucs.louisiana.edu:2373/document/8754067>
* Quantizes to fixed point
* Propose an RTL-designed hardware architecture to accelerate the entire DeepID CNN module on FPGA target.
* Choose DeepID for face feature extraction
* Pre-trained weights and image data are stored on flash
* Coarse grained parallelism is achieved by allowing a multi-channel input map to be used and apply multi-channel weights to create a single output
* Talks about layers of CNN
  + Pooling
  + ReLU (Non-Linear)
  + Fully Connected
    - Has the most data access
* Proposed model is described in Verilog and implemented on Quartus 2. Have implemented a functional simulation

## An Energy-Efficient and Flexible Accelerator based on Reconfigurable Computing for Multiple Deep Convolutional Neural Networks

* **{b5}**
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/8565823>
* Common CNN architecture is not very flexible, this is an issue as the layer size diversity is drastic in larger models
* Propose a Reconfigurable Neural Accelerator (RNA) is designed for adapting to neural network evolution and can easily change CNN shapes like AlexNet, VGG, and Lenet-5.
* Architecture features a spatial array of PE and SPE, FSM, configuration module, and multi-level memory system. The FSM sets configuration parameters for the processing array
  + PE has MAC capabilities while SPE has MAC and non-linear units for bias
  + All hardware is implemented as SPE, but a MUX is used to select PE vs SPE output
* Comments on Eyeriss proposed RS system
  + The RS needs to wait for the previous row of data to be calculated before the next row of image data was read.
* Proposed architecture achieved **good** (?) results on performance and energy efficiency. View table 2 for some data on this
* Still not real reconfigurable hardware

## A software controlled hardware acceleration architecture for image processing using an embedded development board

* **{b7}**
* <https://ezproxyprod.ucs.louisiana.edu:2373/document/7942352>
* Goal is to propose filter hardware that can be controller through software
* Terasic DE2i-150 features CPU and FPGA onboard, CPU communicates with FPGA via PCIe channel onboard
* Image processor was also implemented in software to compare performance
* Device implements Min (erosion) and Max (dilation) filters
* Software allows users to select image, filter, and display the corresponding output feature map
* Software was coded in C/C++
* Uses cascaded register banks to create a series of line buffers essentially
* Still is not what I need for reconfigurable information

## Optimizing the Convolution Operation to Accelerate Deep Neural Networks on FPGA

* {b8}
* <https://ezproxyprod.ucs.louisiana.edu:2373/document/8330049>
* Not reconfigurable
* “More than 90% of the operations in a CNN involve convolutions”
* Set a couple specific goals for optimizing the performance of FPGA convolution operation
  + Computing Latency:
    - Recommend smaller Kernel sizes to reduce the latency
  + Partial Sum Storage / Data Reuse:
    - Focus on calculating partial sums asap and unrolling the algorithm as much as possible
  + Access of On-Chip Buffer:
    - Again, a matter of unrolling the algorithm to reduce the access of data

## Dynamically Reconfigurable Deep Learning for Efficient Video Processing in Smart IoT Systems

* {b10}
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/9221101>
* Omar and Adam’s paper
* Looking to implement DPR on an embedded system platform
* Implement a FINN CNN on the Ultra96-V2 with many configurations by changing weight bit-width and activation bit-width
* Achieves a CNN with s 250MHz clock for all used designs
* “Good case study for the potential of DPR when used in a domain such as constrained IoVT devices.”
* Shows that DPR is good for reducing FPGA device over utilization since parts of the design do not need to be put into hardware and can be load from flash when needed.
* Dynamic Reconfiguration can have different objectives:
  + Behavioral: Change target application
  + Functional: Achieve same application with different internal blocks
  + Trade-Off: Alternating models for power or performance metrics for example

## Energy Adaptive Convolution Neural Network Using Dynamic Partial Reconfiguration

* **{b11}**
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/9184640>
* Develop a CNN architecture to scale accuracy with battery levels
* The goal was to reduce the energy consumption of the CNN as energy reserves deplete without having to completely turn of CNN in device
* “The FPGA reconfiguration time is an important factor in DPR. The reconfiguration time is proportional to the size of the partial bit-stream. As the size of the partial bit-stream increases, the reconfiguration time increases. The size of the partial bit-stream depends on the size of the region to be reconfigured”
* “For adders, registers and comparators, the dynamic power is linearly reduced. While for multipliers, the dynamic power is quadratically reduced.”
* Uses DPR to reconfigure the bit depth calculation which reduces the dynamic power of overall system.
* Achieves 2.7x reduction in energy consumption while only losing 0.53% accuracy for MNIST recognition.

Reconfigurable Real-Time Video Pipelines on SRAM-based FPGAs

* **{b12}**
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/8994814>
* Using pynq API for PR video pipelining

A Reconfigurable Streaming Deep Convolutional Neural Network Accelerator for Internet of Things

* {b13}
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/8011462>
* Probably better to describe the design as “configurable”
* Implemented on ASIC
* Optimizes for energy efficiency

**Analyzing the Energy-Efficiency of Vision Kernels on Embedded CPU, GPU and FPGA Platforms**

* {b14}
* <https://ezproxyprod.ucs.louisiana.edu:2373/document/8735546>
* Analyzes the power efficiency of GPU and FPGA and benchmark a couple processes compared to a CPU

# Old Papers

## Dynamic Partial Reconfiguration in FPGAs (2009)

* {o1}
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/5369525>
* Early investigation (2009) for the use of DPR on FPGA
* **Difference -based** partial reconfiguration is used when there is a small change to the overall design. Tracks just the difference in the PR region and makes those changes only. Maybe changing coefficients to a block or equations for a LUT
* **Module-based** partial reconfiguration uses modular design concepts to reconfigure large blocks of logic. This is what is more common in DPR today. Reconfigurable blocks can be defined to be interchanged in reconfigurable regions

## Efficient FPGA implementation of convolution (2009)

* **{o2}**
* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/5346737>
* As the name suggests, it looking to implement an efficient convolution processor on FPGA as other models at the time (2009) were very application specific
* Comments on the use of FFT algorithms. Claims that since they rely on counters and RAM blocks, the activity factor increases which would negatively affect power consumption
* Not too much to take away considering the paper is from 2009 and is before a lot of DPR advancements occur which can really benefit convolution
* Good information on the general process of implementing a convolution on FPGA

# Just some links, not really looked at

## Partial reconfiguration on FPGAs in practice — Tools and applications

* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/6222217>

## Integrated Optimization of Partitioning, Scheduling, and Floorplanning for Partially Dynamically Reconfigurable Systems

* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/8552457>

## Reconfigurable Convolution Architecture for Heterogeneous Systems-on-Chip

* <https://ieeexplore-ieee-org.ezproxyprod.ucs.louisiana.edu/document/9134344>

## ImageNet classification with deep convolutional neural networks

* <https://dl.acm.org/doi/abs/10.1145/3065386>