FPGA Capestone

Paper Presentation

Fpga-based SoC design for real-time facial point detection using deep convolutional neural networks with dynamic partial reconfiguration

Safa Teboulbi1 · Seifeddine Messaoud2 · Mohamed Ali Hajjaji3 · Abdellatif Mtibaa4 · Mohamed Atri5

Journal: Signal, Image and Video Processing
Published online: 14 May 2024
https://link.springer.com/article/10.1007/s11760-024-03177-2

Introduction

- This paper proposes a DPR based approach for deep-CNN for facial key point detection while comparing it with other hardware.
- Approaches :-
  - GPU Based Implementation
  - CNN acceleration & optimization using HLS
  - DPR based hybrid architecture
- Target was to achieve better performance in terms of execution time, hardware cost and power consumption.

Motivation

- DCNNs need more accuracy than standard ANN algorithms. Therefore, they need enormous memory access and computational resource which represents a significant challenge for CPUs and absorbs a huge amount of power.
- Due to large number of layers and the compute complexity these algorithms are difficult to implement on a embedded system.
- To address this issue GPUs, DSPs, ASICs & FPGAs are being adopted.

GPU & HLS approach

- F1 network was implemented on a GPU which achieved 89.01%, precision of 91.63%, and recall of 90.25%
- The power consumption in GPU was very high.
- Hardware acceleration using HLS significantly improved execution times for CNVL layers, with the second CNVL layer being accelerated by 45.01 times and convolution 1 by 41.8 times compared to software.

Multi-FPGA cluster

- The PYNQ boards are being used in this cluster as they are economical Zynq based boards.
- Implementation of ResNet-50 on a 4-FPGA cluster showed performance of 75FPS throughput & 292 GOPS performance.
- This performance was much faster than CPUs and much more efficient than GPUs

Multi-FPGA cluster

- The PYNQ boards are being used in this cluster as they are economical Zynq based boards.
- Implementation of ResNet-50 on a 4-FPGA cluster showed performance of 75FPS throughput & 292 GOPS performance.
- This performance was much faster than CPUs and much more efficient than GPUs

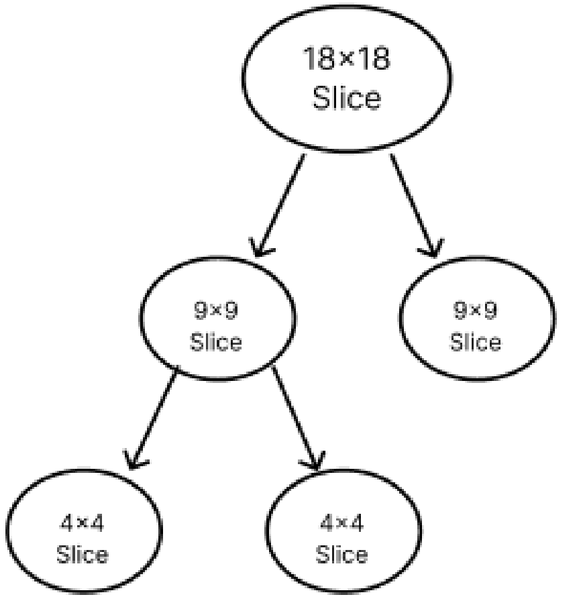
Cascaded Multi-layer perceptron NN

- Cascading low-end FPGAs to split the workload. Hand Gesture Recognition(HGR) was implemented on 2 Spartan Boards.

- System could detect 5 hand gestures with 97.3% accuracy.

Bit-width partitioning

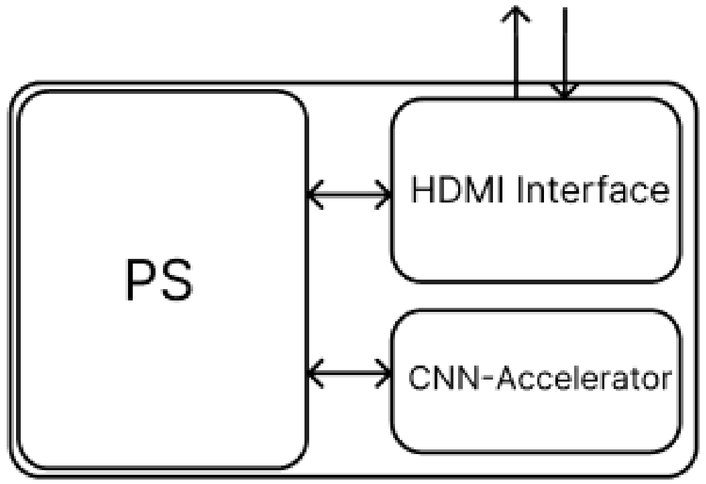
- Assuming a FPGA where the DSP slice by default does 18x18 processing, in some cases we do not require 18 bits where by default rest bits are set as zeros which results in in-efficient processing and poor utilization of resources.
- Bit-width partitioning is where a DSP slice is split into multiple slices to process on smaller data sizes.
- So a 18x18 slice can be split into 2 9x9 slices and further down to 4 4x4 slices.



HLS Based approach

- Implemented on Pynq-Z1 MPSoC FPGA. Where 2 HDMI ports are configured for Video input and output through the PL.

- Processing system of the SOC receives the image through the HDMI port via a DMA buffer.
- The PS does preprocessing where the Faces are detected, designated with a bounding box, and cropped from the picture in the first phase. Therefore, each image.
- This method is very promising but the resource utilization is very high and also power consumption is high

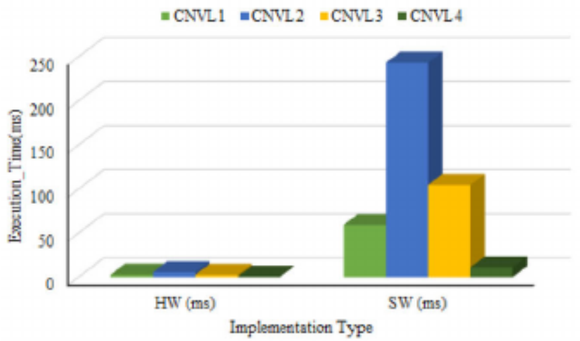


HLS Based approach

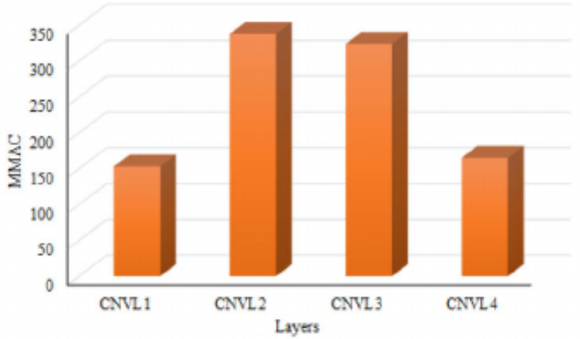
- Benchmarks

CNVL1, CNVL2, and CNVL3 need the highest time to calculate all dot products, which is about 243.1 ms in SW to 5.4 ms in HW, 59 ms in SW to 2.6 ms in HW, and 104.5 ms in SW to 2.5 ms in HW, respectively.

Layer	DSP	BRAM	FF	LUTs
CNVL 1	29%	24%	24%	34%
CNVL 2	20%	21%	17%	31%
CNVL 3	20%	21%	24%	28%
CNVL 4	7%	24%	9%	18%



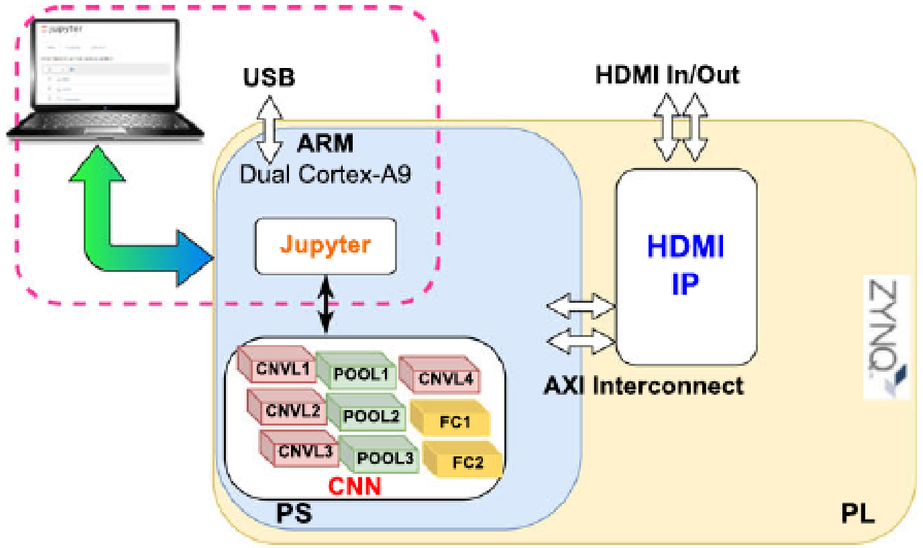
(c) Execution time of the CNVL IPs in HW and in SW



(d) Mega Multiply-Accumulate Operation per second

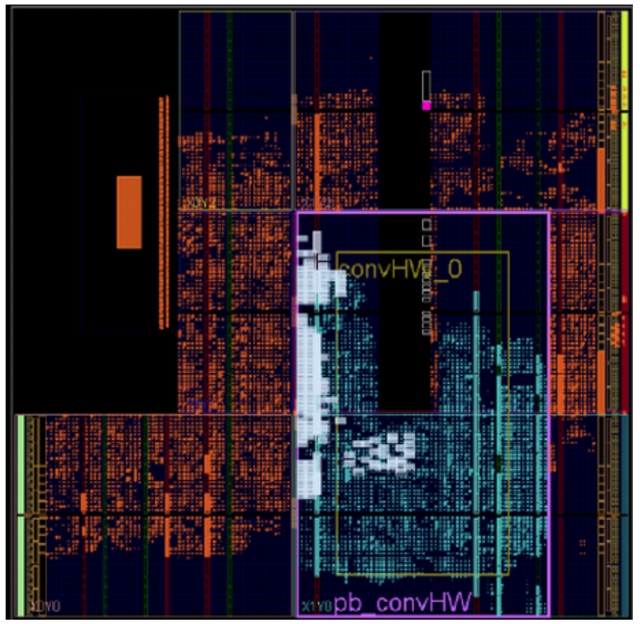
Software Based Approach

- In this approach the whole convolution was run on the PS of the PYNQ board and the PL was used only as a HDMI interface. This showed a execution time of 440mS.



Dynamic Partial Reconfig technique

- In Dynamic Partial Reconfiguration the FPGA die is divided into static and dynamic region, During run time the dynamic region can be reconfigured without affecting other Static/Dynamic regions on the die.
- The PYNQ SoC which is Zynq 7020 does not have a on-chip HDMI interface therefore the PL part of SoC is configured to behave as the interface which utilizes a major part of the PL. The Left over region that is 28% of FF, 30% of BRAM, 40% of LUT, and 35% DSP is configured are Reconfigurable CLB.
- The Convolution is split into 4 IPs for reconfiguration , and is observed that CNVL2 is the most compute intensive.



Dynamic Partial Reconfig technique

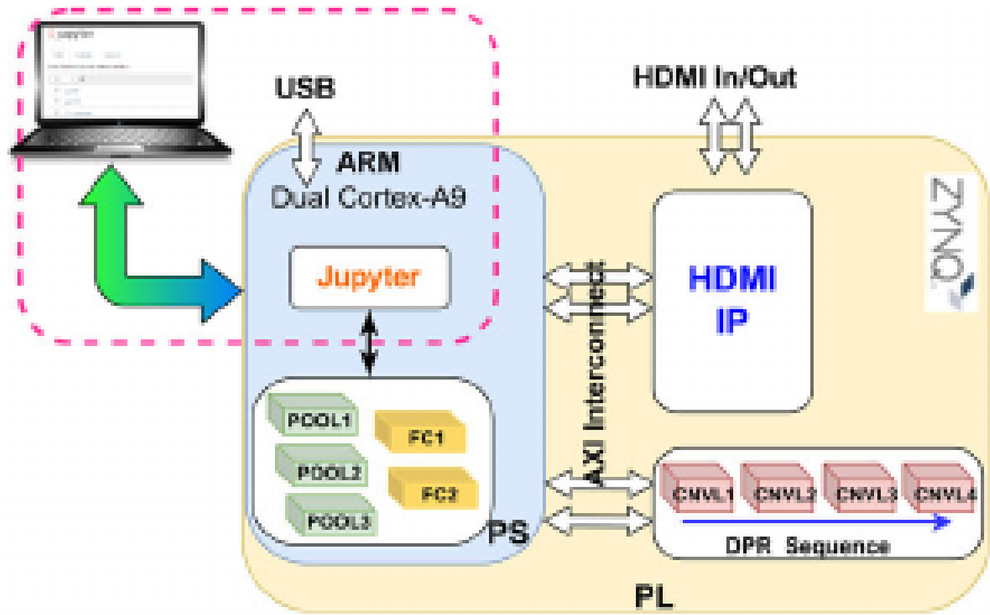
Observation

- In this approach final benchmarks are worse than the software implementation(CPU) which is about 440mS.
- The results showed execution time exceeding 500mS.
- Though the CNVL IP has a mean computation time of 2.6ms. Majority of the time is spent on reconfiguration. whose is
  mean is 137ms hence resulting is
  high computation time.

Dynamic Partial Reconfig technique

Observation

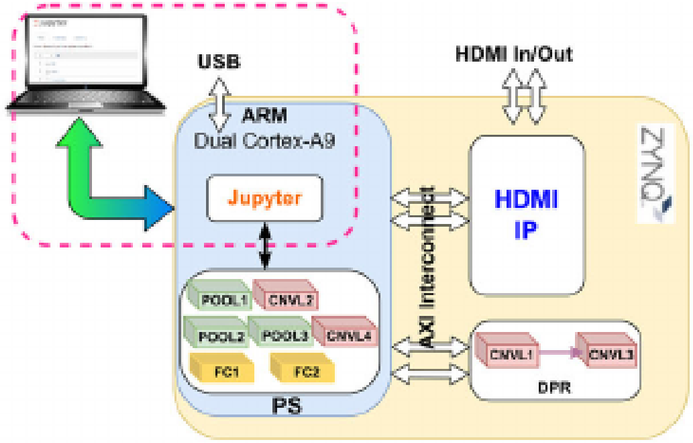
- In this approach final benchmarks are worse than the software implementation(CPU) which is about 440mS.
- The results showed execution time exceeding 500mS.
- Though the CNVL IP has a mean computation time of 2.6ms. Majority of the time is spent on reconfiguration. whose is mean is 137ms hence resulting is high computation time.

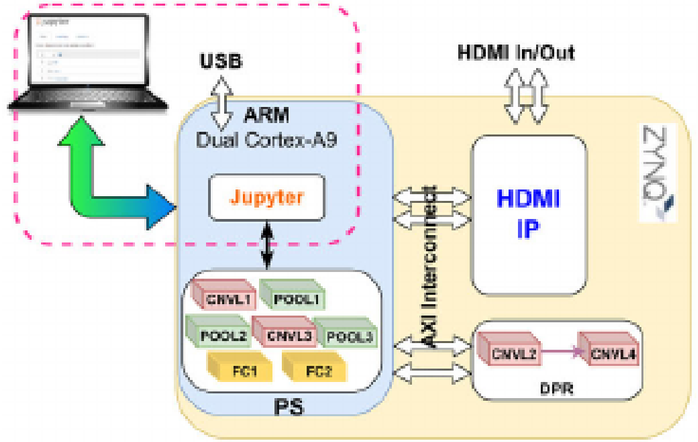


Dynamic Partial Reconfig technique

DPR Optimization

- Based on Rconfig-time and complexity of the layer they are moved into PS & PL.
- Strategy 1 CNVL1 IP & CNVL3 are moved into PL while CNVL2 & CNVL4 are in PS. This acceleration can speed up the software run time by 1.5 times.
- Strategy 2 CNVL2 IP & CNVL4 are moved into PL while CNVL1 & CNV34 are in PS. This acceleration can speed up the software run time by 2.3 times.





Dynamic Partial Reconfig technique

Final Results

- The obtained results showed that the proposed approach achieves higher performance in terms of reconfiguration time, execution time, hardware cost, and power consumption in comparison with other state-of-the-art implementations on FPGAs.