UNIVERSITY OF ARKANSAS

Accelerating DICe on FPGA

Atiyehsadat Panahi - PhD Student

Advisor: Dr. David Andrews

March 27th, 2018

Objective of this research

- How appropriate are FPGAs to accelerate DICe?
  - How much speedup over a workstation
- Is HLS a better approach than HDL?
  - What performance effects occur to gain productivity of HLS
  - Can we create reusable libraries to increase future productivity
- How much acceleration can we get using HMC?
  - Can the multichannels between 3D stacked memory and FPGA be leveraged for increasing performance of DICe computations?
- *Can reuse of FPGA system be increased through MPSoPC Architecture ? (longer term)

Approach

- Install/ Run DICe on workstation
- Understand the architecture of DICe
- Integrate FPGA with workstation
- Rewrite kernels

Status

- Just starting ☺
- Installed DICe
- Studied software architecture
  - How the correlation is done
  - The evaluation criteria, input and output params
  - The bottleneck has been found in two “for” loops in objective functions
- Looking at how to get images/frames to DRAM/BRAM
  - Successful image reads and writes into FPGA using BRAM

Technical Issues

- Loading the text files into the FPGA
  - Not all the input files are images (subsets.txt, params.xml and input.xml)
- Input data size
  - Input size = # frames × frame size = 15828 × 640 × 480 × 1B = 4.52 GB
  - BRAM size = 4.5 MB ➔ just 15 frames could be stored in BRAM
  - DRAM size = 1 GB ➔ $\frac{1}{5}$ of the frames could be stored in DRAM
  - We have to use SD Card!
- License problem with SD controller

UNIVERSITY OF ARKANSAS

Next (Interim) Steps

- Divide and Conquer to minimize complexity
  - Accelerating kernel in FPGA
  - Forming fast efficient network connection
- Keeping it simple to start: Accelerating Kernels in FPGA
  - Implementing what DICe does just on two frames for the start
  - Trying other possible solutions for reading the input data (maybe UART)
  - Reading and writing the images into SD card
- Will allow us to explore the use HLS and HDL to accelerate the two main “for loops”
- Network Interfacing:
  - After accelerating “for loops” address network connection between PC and FPGA.