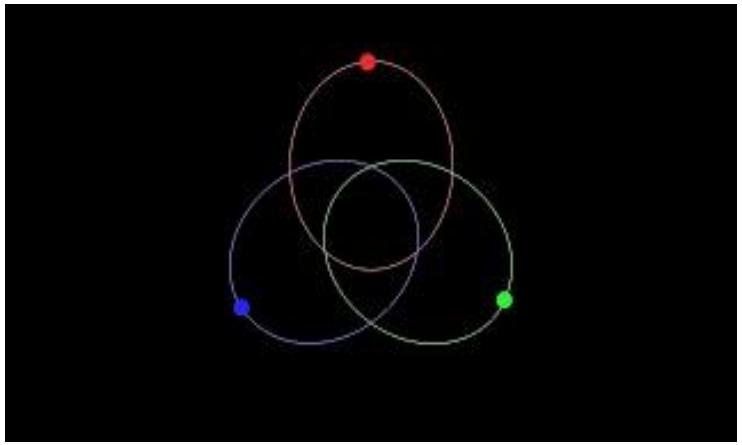
# Team A3: Accelerated N-Body Simulations

Abhinav Gupta, Yuhe Zheng and Rene Ravanan

# Background

**Faster N Body simulation**

- N-body simulations are a type of simulation that model the behavior of a physical system using a set of discrete entities.
- These simulations are typically time dependent and are used to study astronomical systems of gravitationally interacting subunits.



# Use Case and Motivation

- **Problem:** For physicists the N Body simulation is an important and computationally hard problem to solve, trying to run the algorithm parallely on a CPU is simply not fast enough, and running on GPU is not power efficient
- **Solution:** Run the N Body simulation on a FPGA and try to achieve a 10x speedup with ~2x energy efficiency
- **ECE Areas:** Software Systems, Hardware Systems

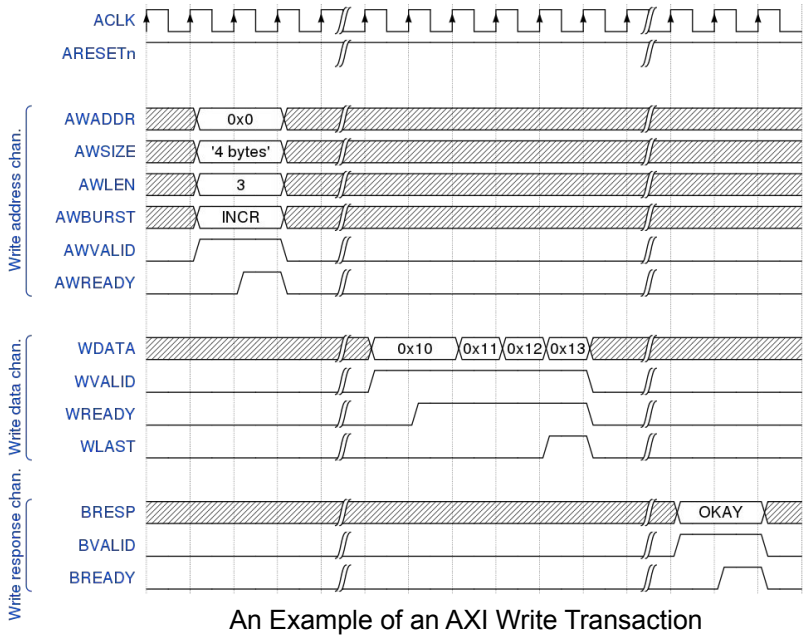
# Use-Case Requirements

- **Usability:**
  - Accelerated implementation should be easy enough to use (i.e. complications in the user experience should not outweigh our performance gain)
- **Speedup**
  - We are planning on achieving a **10x speedup** as compared to a CPU
- **Energy efficiency**
  - Average CPU TDP is between 50W-100W
  - We aim to achieve a ~20W TDP envelope

# Data Communication Requirements

- Communication between the CPU and FPGA should not outweigh computation process.
- Data transfer between global memory, host program, and computation kernel should not bottleneck simulation.

(We also considered an ethernet based set up, but this would have been out of scope).




An Example of an AXI Write Transaction

# Technical Challenges

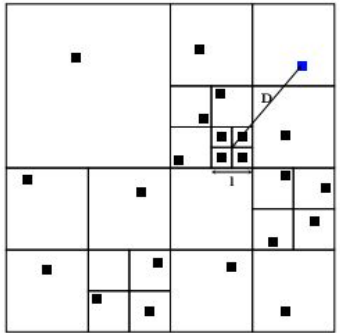
- Working with Vitis:
  - Leveraging Control and Data Driven TLP (Task Level Parallelism) in the most optimal way.
- Irregularity in Computation:
  - Requires a good workload distribution
- SW/HW interfacing.
  - Communication to/from memory has to be seamless and should not bottleneck the actual computation.

# Solution Motivation

Why FPGA?

- Disadvantage of CPU
  - CPU would be hardware limited with respect to the number of threads it can create due to the large particle number
- Disadvantage of GPU
  - GPU suffers both kernel irregularities and is quite energy inefficient
- Advantages of FPGA:
  - Exploit spacial concurrency while performing irregular computations
  - Remove CPU overhead for supporting general purpose applications
  - Flexible memory structure to optimize computation and data throughput

# Solution Algorithm (Barnes Hut vs All Pairs Approach)



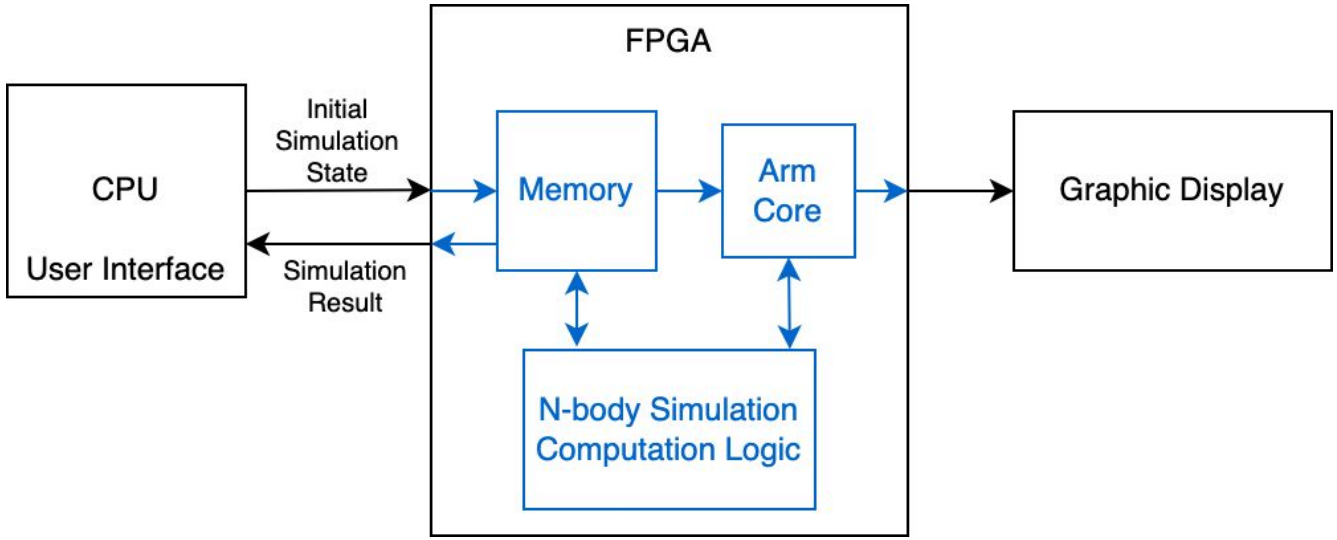

Quad-Tree Used by Barne's Hut

```
1: procedure N-BODY
2:     for each time step t do
3:         for each Body i do
4:             for each Body j, j ≠ i do
5:                 PairwiseForce(i, j)
6:             end for
7:             UpdateBody(i)
8:         end for
9:     end for
10: end procedure
```

All Pairs Approach

- All pairs is a more fundamental approach but the time complexity is O(N^2).
- Barnes Hut is O(NlogN) but it is algorithmically harder to implement for example trying to resize the quad trees and doing load balancing is a major part of parallelizing Barnes Hut.This can prove to be challenging on an FPGA
- Parallelizing the all pairs algorithm is simpler but it is also generally slower than barnes hut.

# Solution Approach




- FPGA computes N-body simulation and drive real-time graphic display
- CPU takes user input to initialize simulation data and receives simulation result

# Tool Chains

- **Vitis Platform:**
  - Host Program Compilation
  - High Level Synthesis
  - Utilization Report Collection

- **Ultra96-V2 FPGA:**
  - Arm Core
  - Programmable Logic

# Testing, Verification, and Metrics

- **Design Simulation:**
  - Performing Vitis software/hardware emulation to verify system behavior and logic functionality
  - Using JTAG to verify board behavior
- **Correctness Evaluation:**
  - First verify that both sequential and parallel implementations produce the correct results
  - This will be done by having a precomputed expected result for each simulation.
- **Speedup:**
  - Measuring time taken to execute our solution and comparing that with our CPU benchmark.

# Division of Labour

Abhinav:

- Identifying and setting up appropriate display hardware
- Refactor Algorithm for FPGA Translation
- Introducing UI
- Set up HDMI Interface for Visualisation

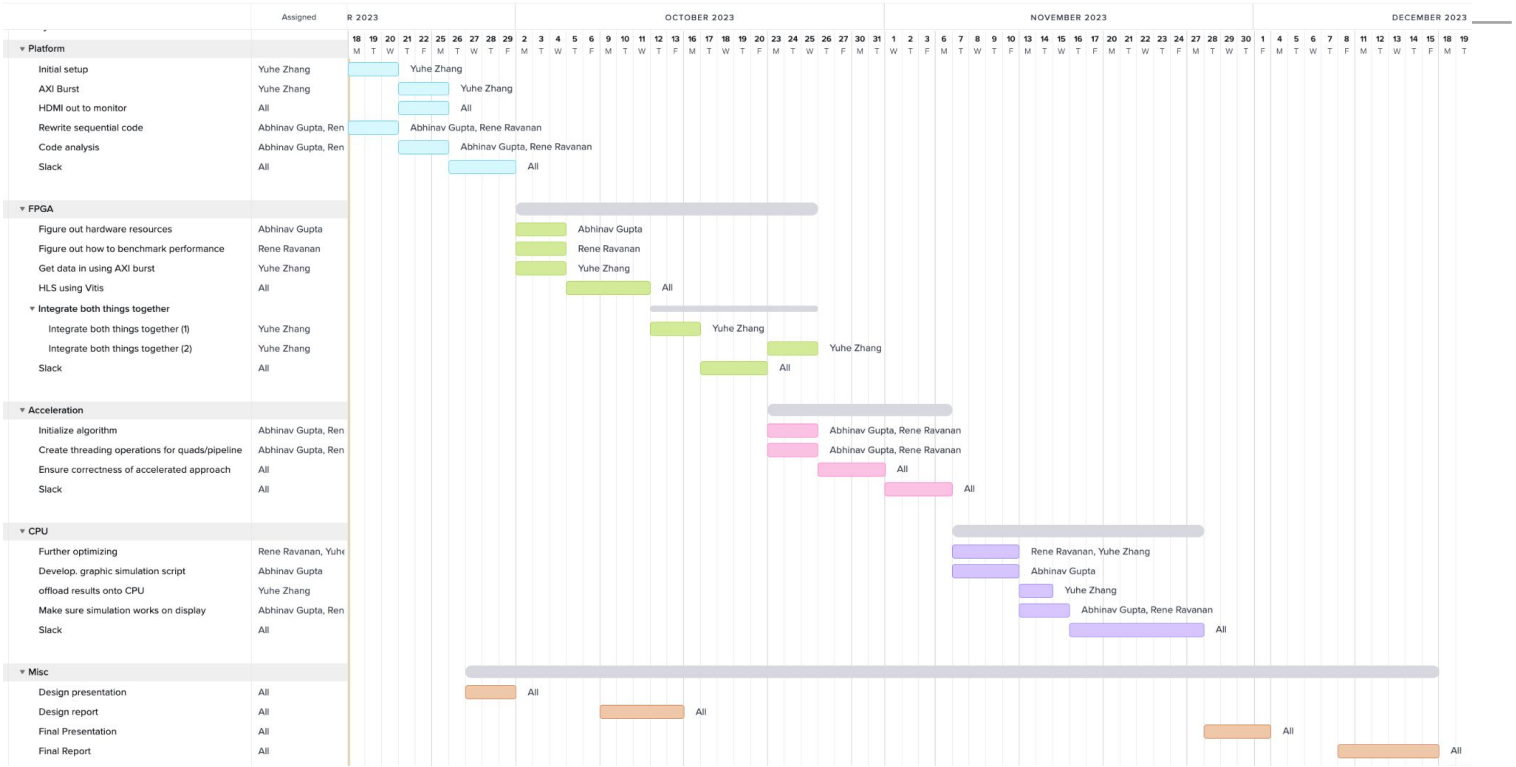
Rene:

- Analyse computational kernel for parallelism/acceleration opportunities
- Refactor Algorithm for FPGA Translation
- Optimise N-body simulation algorithm on FPGA

Yuhe:

- Manage Data Transfer Between FPGA and CPU
- Setup VITIS HLS workspaces and build configurations
- Hardware Pipeline Integration
- Optimise N-body simulation algorithm on FPGA

# Schedule


Assigned
R 2023
OCTOBER 2023
NOVEMBER 2023
DECEMBER 2023
Platform
Initial setup
Yuhe Zhang
AXI Burst
Yuhe Zhang
HDMI out to monitor
All
Rewrite sequential code
Abhinav Gupta, Ren
Abhinav Gupta, Rene Ravanan
Code analysis
Abhinav Gupta, Ren
Abhinav Gupta, Rene Ravanan
Slack
All
FPGA
Figure out hardware resources
Abhinav Gupta
Figure out how to benchmark performance
Rene Ravanan
Get data in using AXI burst
Yuhe Zhang
HLS using Vitis
All
Integrate both things together
Integrate both things together (1)
Yuhe Zhang
Integrate both things together (2)
Yuhe Zhang
Slack
All
Acceleration
Initialize algorithm
Abhinav Gupta, Ren
Abhinav Gupta, Rene Ravanan
Create threading operations for quads/pipeline
Abhinav Gupta, Ren
Abhinav Gupta, Rene Ravanan
Ensure correctness of accelerated approach
All
Slack
All
CPU
Further optimizing
Rene Ravanan, Yuhe
Rene Ravanan, Yuhe Zhang
Develop. graphic simulation script
Abhinav Gupta
Abhinav Gupta
offload results onto CPU
Yuhe Zhang
Yuhe Zhang
Make sure simulation works on display
Abhinav Gupta, Ren
Abhinav Gupta, Rene Ravanan
Slack
All
All
Misc
Design presentation
All
All
Design report
All
All
Final Presentation
All
All
Final Report
All
All