

# UPIC: A Framework for Massively Parallel PIC codes

Viktor K. Decyk

UCLA

Los Angeles, California, USA

Why do we need a software framework?

- Computer architecture is becoming more complex
- Physics models are becoming more ambitious
- Parallel programming requires special expertise
- Team efforts are becoming necessary.

It takes too long for a graduate student to become familiar with and modify codes.

3D PIC is particularly challenging:  
requires high performance and parallel computers

# UPIC Framework

Framework: a unified environment containing all components needed for writing code for a specific problem domain

Goal is rapid construction of new codes by reusing tested modules: “Lego” pieces for developing new codes

Designed to support different programming styles

- Simple Fortran77 projects
- Complex, object-oriented, multi-author projects
- Student programmers, with many error checks

Supports multiple numerical methods, optimizations, different physics models, different types of hardware

Above all, hides parallel processing

## Multiple Electromagnetic Models

- Electrostatic (Coulomb force)
- Darwin (induced electric and magnetic forces)
- Electromagnetic (Maxwell's equation)
- Relativity

## Boundary conditions

- Periodic, Dirichlet, Neumann, Open (vacuum)

## Multiple levels of accuracy

- Linear, Quadratic interpolation, Gridless
- Single, Double precision

## Multiple programming paradigms

- Threads
- MPI

Currently, based on Spectral Methods

- High accuracy
- Energy, Momentum conservation very good
- Useful for evaluating alternate algorithms => Fake Physics
- Useful when comparing with plasma theory

Finite-difference, non-uniform meshes to be added, evaluated

V. K. Decyk, "UPIC: A framework for massively parallel particle-in-cell codes," Computer Phys. Comm. 177, 95 (2007)

# Layered Approach

Lowest Layer: optimized Fortran 77 routines

- complex, unsafe, but very fast
- can be called by many languages

Middle Layer: Fortran 90 wrappers to lowest layer

- Simpler usage by hiding information, type checking
- Dynamic memory, pointers
- Familiar procedure style
- Can be written in other languages, C++, Fortran2003

Upper Layer: Object-Oriented Design Patterns

- Large blocks of code can be reused as black boxes
- Separation of concerns useful with multiple authors

# Parallelization of PIC Code

Domain decomposition for distributed memory computers

Primary Decomposition load balances particles

- Sort particles according to spatial location
- Same number of particles in each non-uniform domain
- Scales to thousands of processors

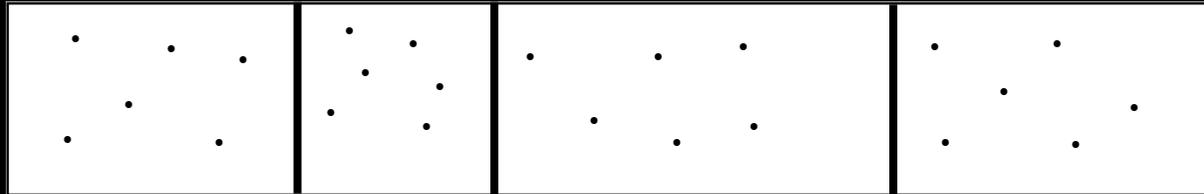
Particle Manager responsible for moving particles

- Particles can move across multiple nodes

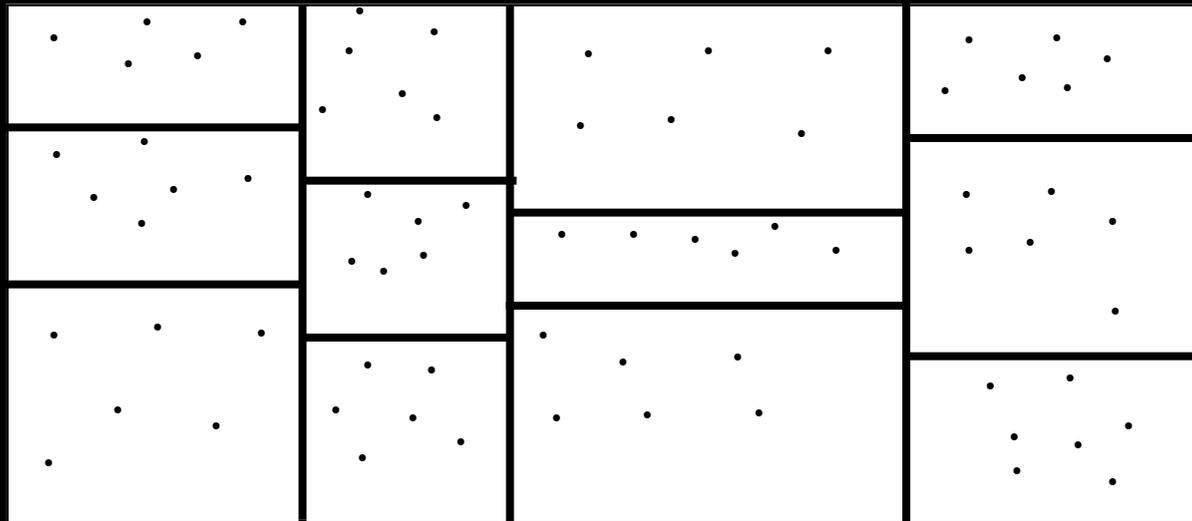
Uniform Secondary Decomposition to balance field solver

Partition Manager moves data between partitions

- Fields can move across multiple nodes



**1D Domain decomposition**

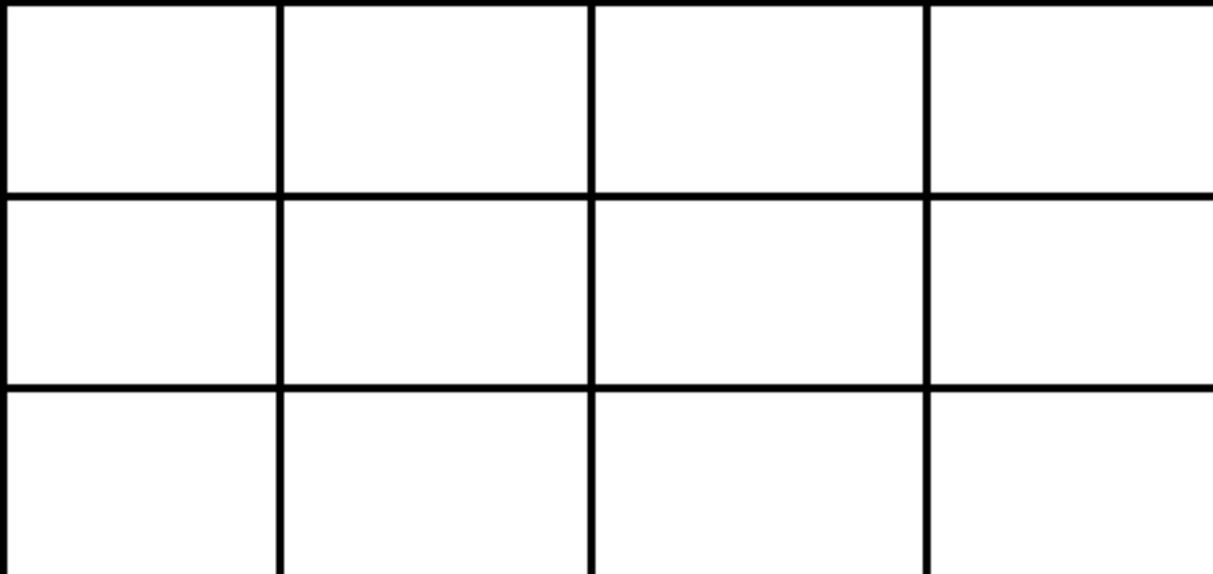


**2D Domain decomposition**

**Each partition has equal number of particles**

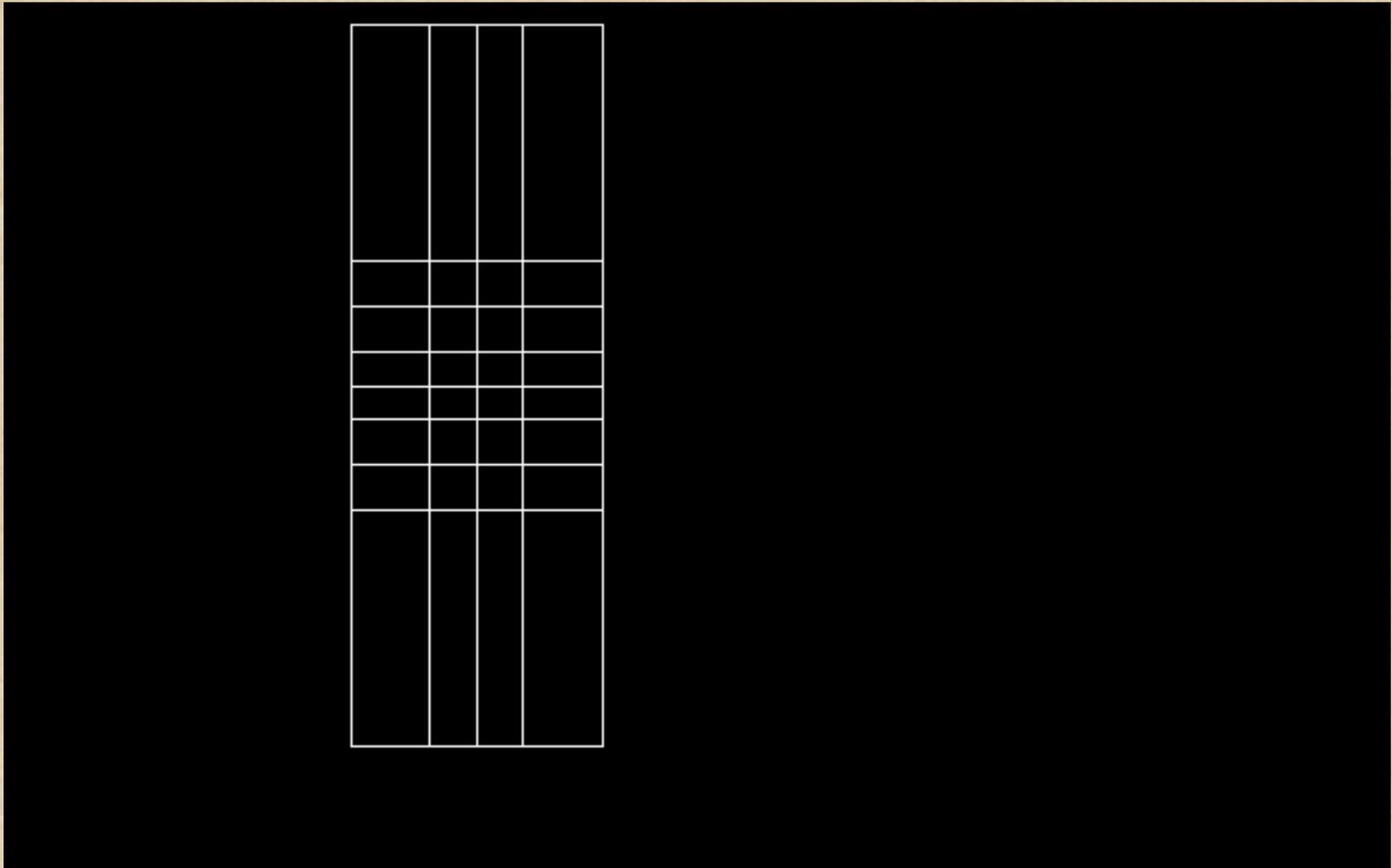


**1D Domain decomposition**

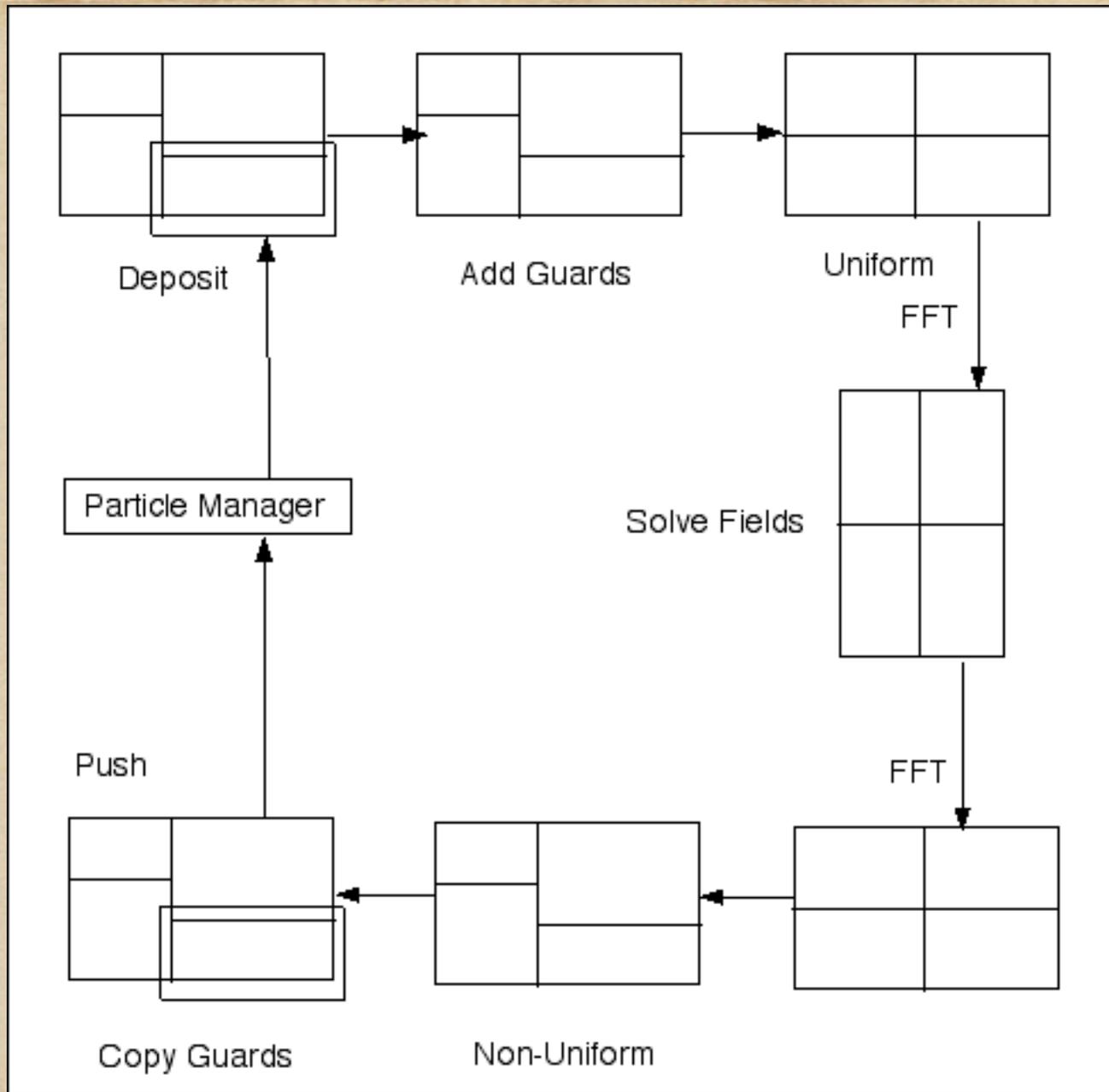


**2D Domain decomposition**

**Each partition has equal number of grids**



Dynamic load balancing: changing partitions



Field Data Distributions in Main iteration loop

## PLIB: Parallel Particle Simulation Library

A small low-level library (30 subroutines) which encapsulate all the communication patterns needed by parallel PIC codes

- designed for high-performance
- hides low-level communication details
- supports both message-passing and shared memory
- supports RISC and vector architecture
- supports linear and quadratic interpolation

Used in several “Grand-Challenge” Calculations

- Numerical Tokamak Turbulence Project
- Space Simulation
- Advanced Accelerators

Written in Fortran77: fast, can be called from many languages

Single node performance, 2.5 GHz Macintosh G5

2D Electrostatic, 45 nsec/particle

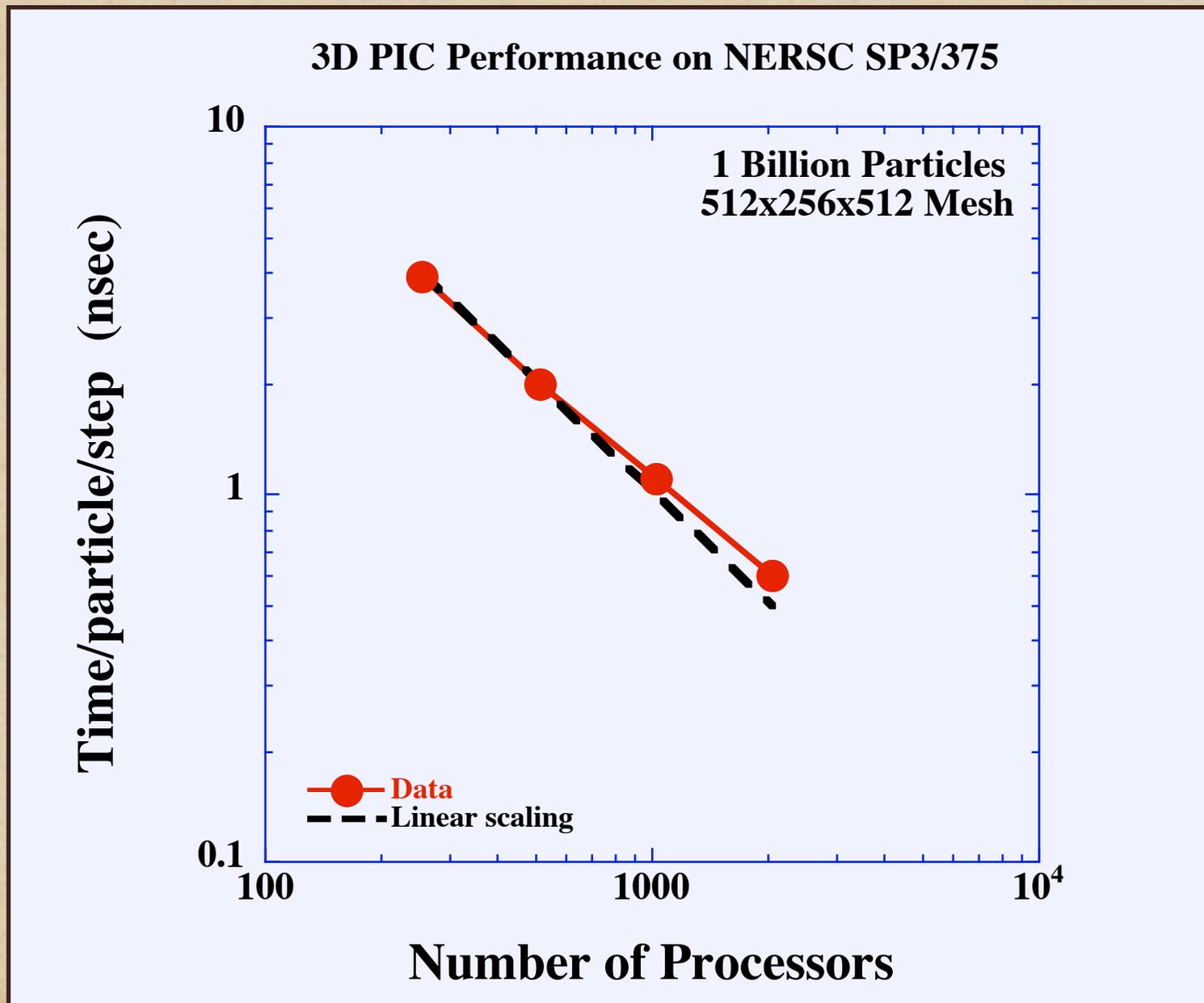
2-1/2D Electromagnetic, Relativistic, 215 nsec/particle

2-1/2D Darwin, 410 nsec/particle

3D Electrostatic, 85 nsec/particle

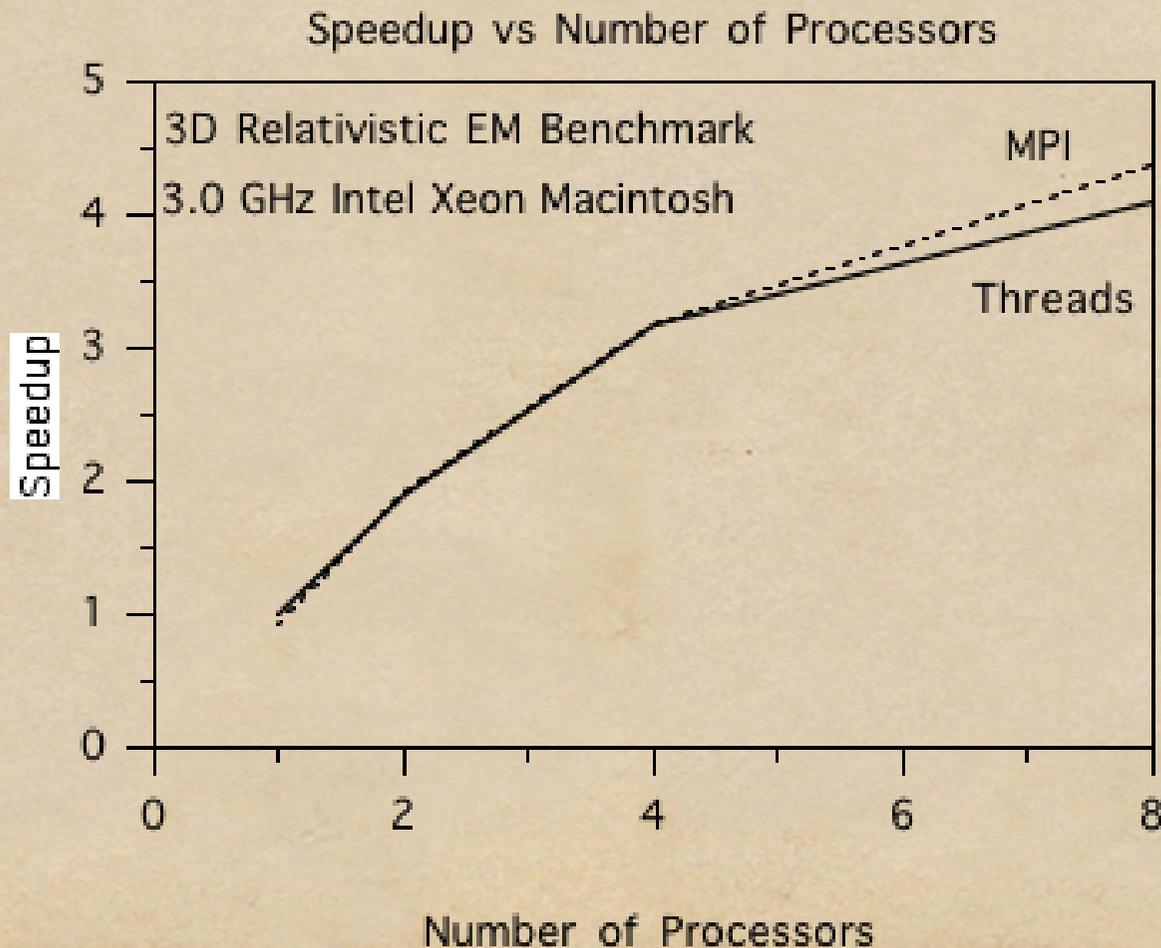
3D Electromagnetic, Relativistic, 270 nsec/particle

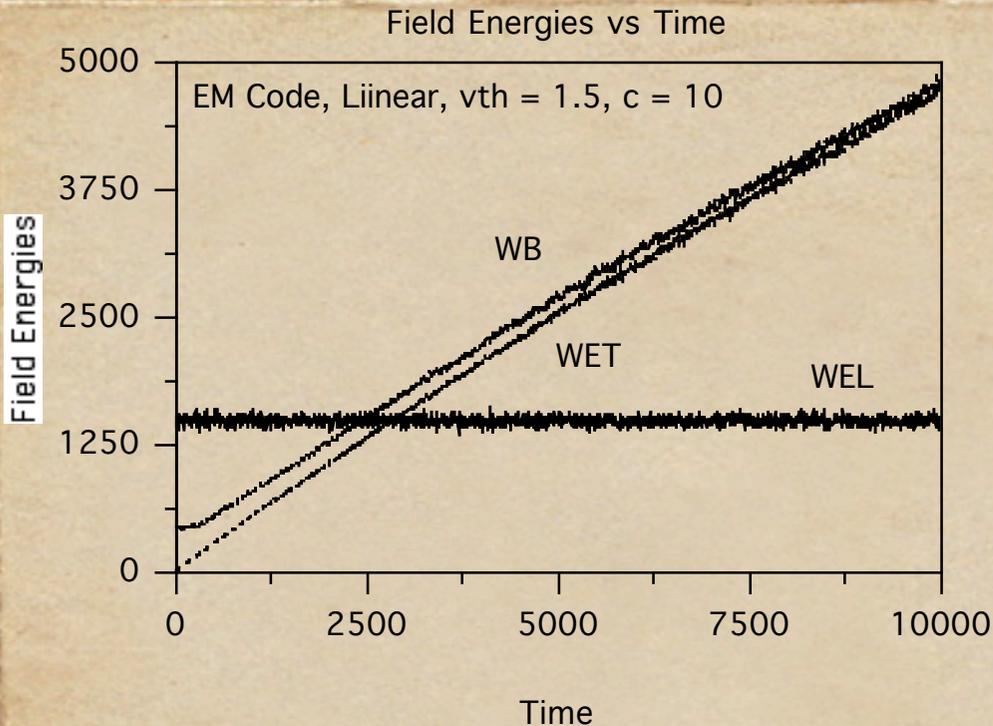
# Scaling of Large PIC Electrostatic Benchmark



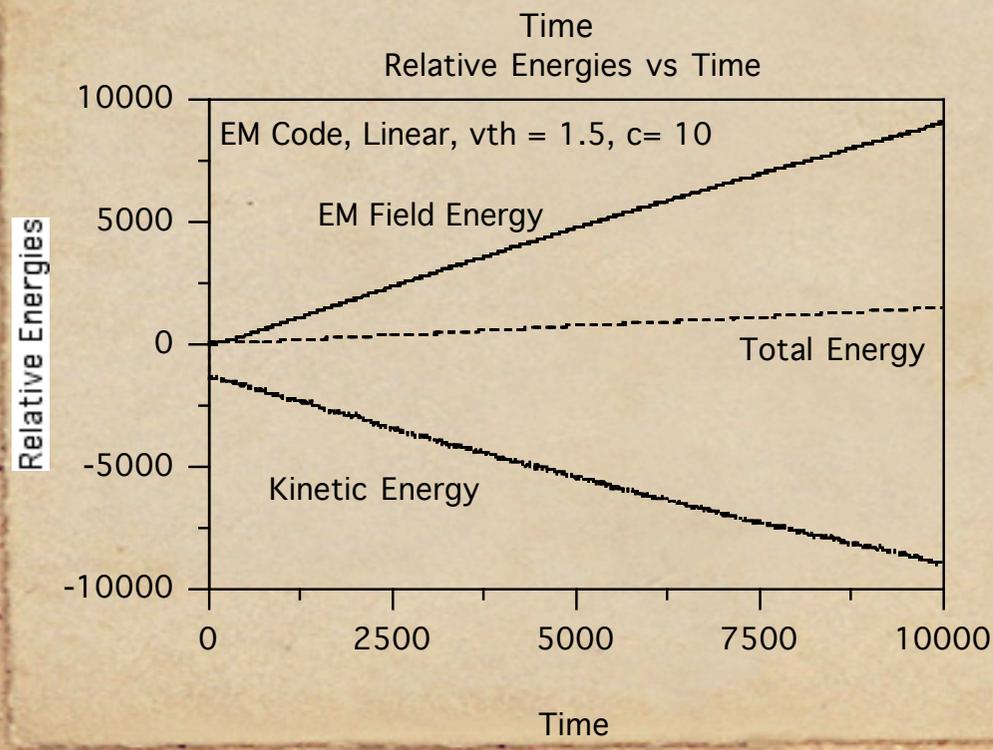
# Future Architectures

Will multicore processors change our strategy?  
Cache coherency?

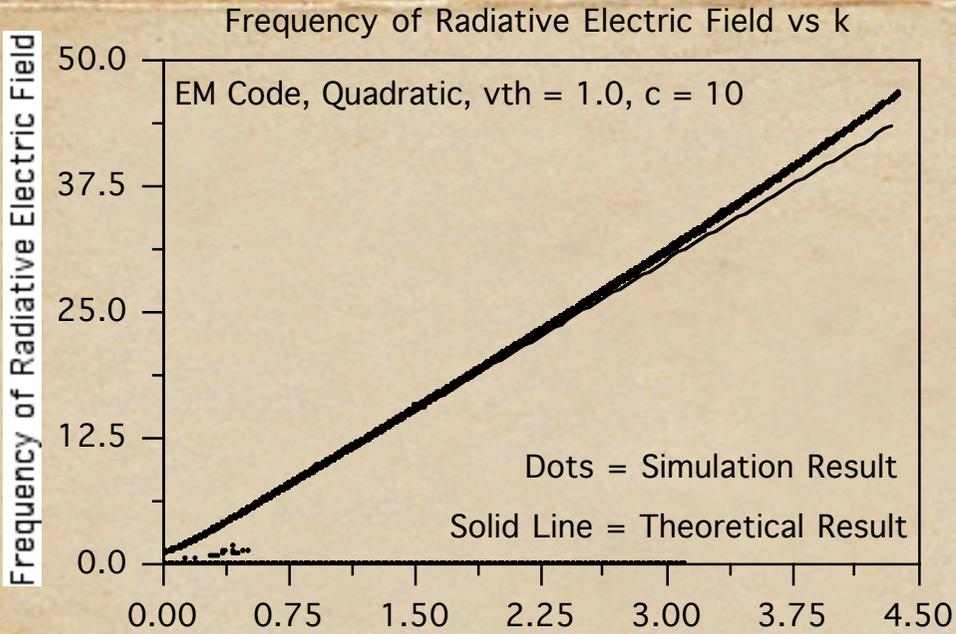




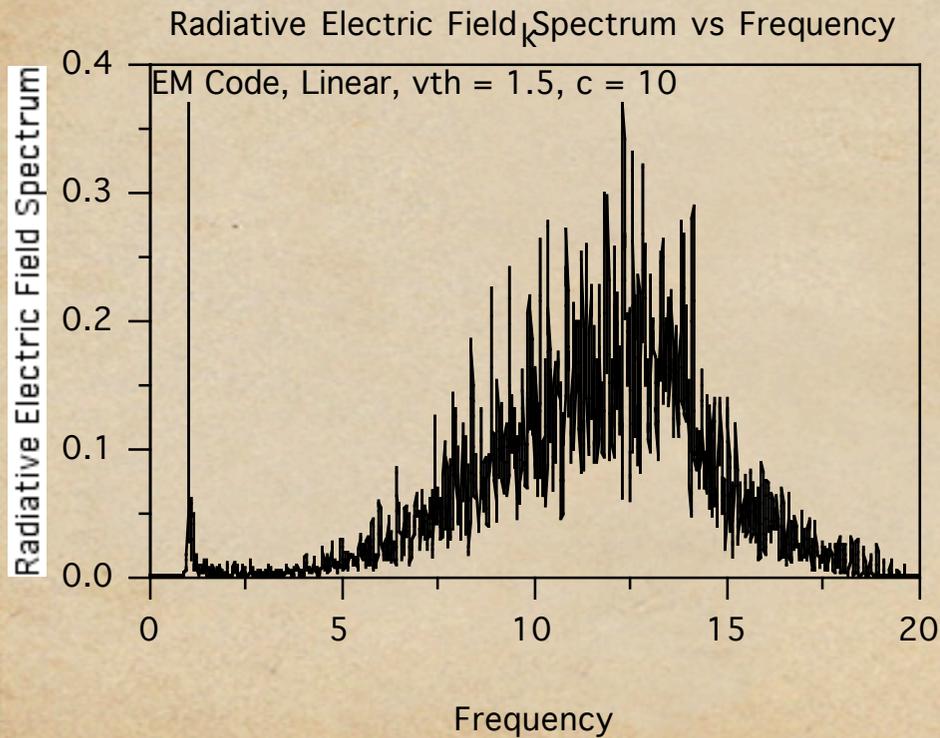
In a plasma in “thermal equilibrium”, transverse electromagnetic energy grows without bounds. Is this real?



This energy growth comes from particle kinetic energy and nearly conserves energy.

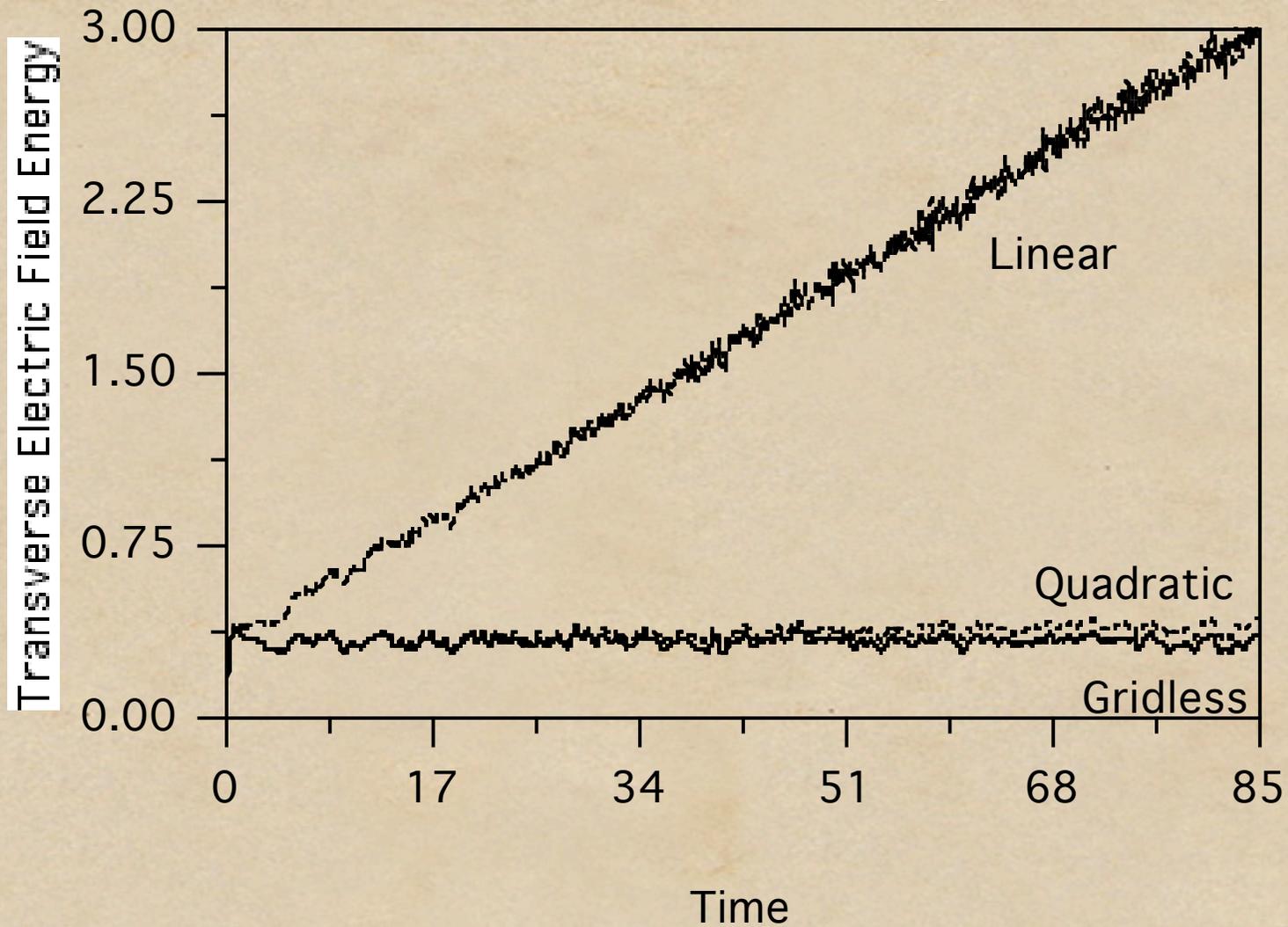


Dispersion relation for light waves looks OK.



But there is a lot of energy at high frequencies!

Transverse Electric Field Energy vs Time



Varying precision shows this is a grid effect. It is not real!

# Design Patterns

Complex software requires more concepts than just objects

Design patterns are language independent abstractions to organize classes to solve recurring software problems.

- Design patterns gives a way to reason about organizing complex projects

In collaboration with Henry Gardner, ANU, we are investigating useful patterns for scientific programming and translating them to Fortran95.

# Main simulation program in Fortran 95: facade pattern

```
program simulation2d
use plasma2d_class
integer :: done = 0
type (plasma2d) :: plasma

call new_plasma2d(plasma)
do while (done >= 0)
  done = update_plasma(plasma)
enddo
call del_plasma(plasma)
```

# Main simulation program in C

```
int main(int argc, char *argv[])
{
    int itime = 0, done = 0;
    int plasma;

    NEW_2DPLASMA(&plasma);
    while (done==0) {
        UPDATE_2DPLASMA(&plasma,&itime,&done); }
    DEL_2DPLASMA(&plasma);
    return 0;
}
```

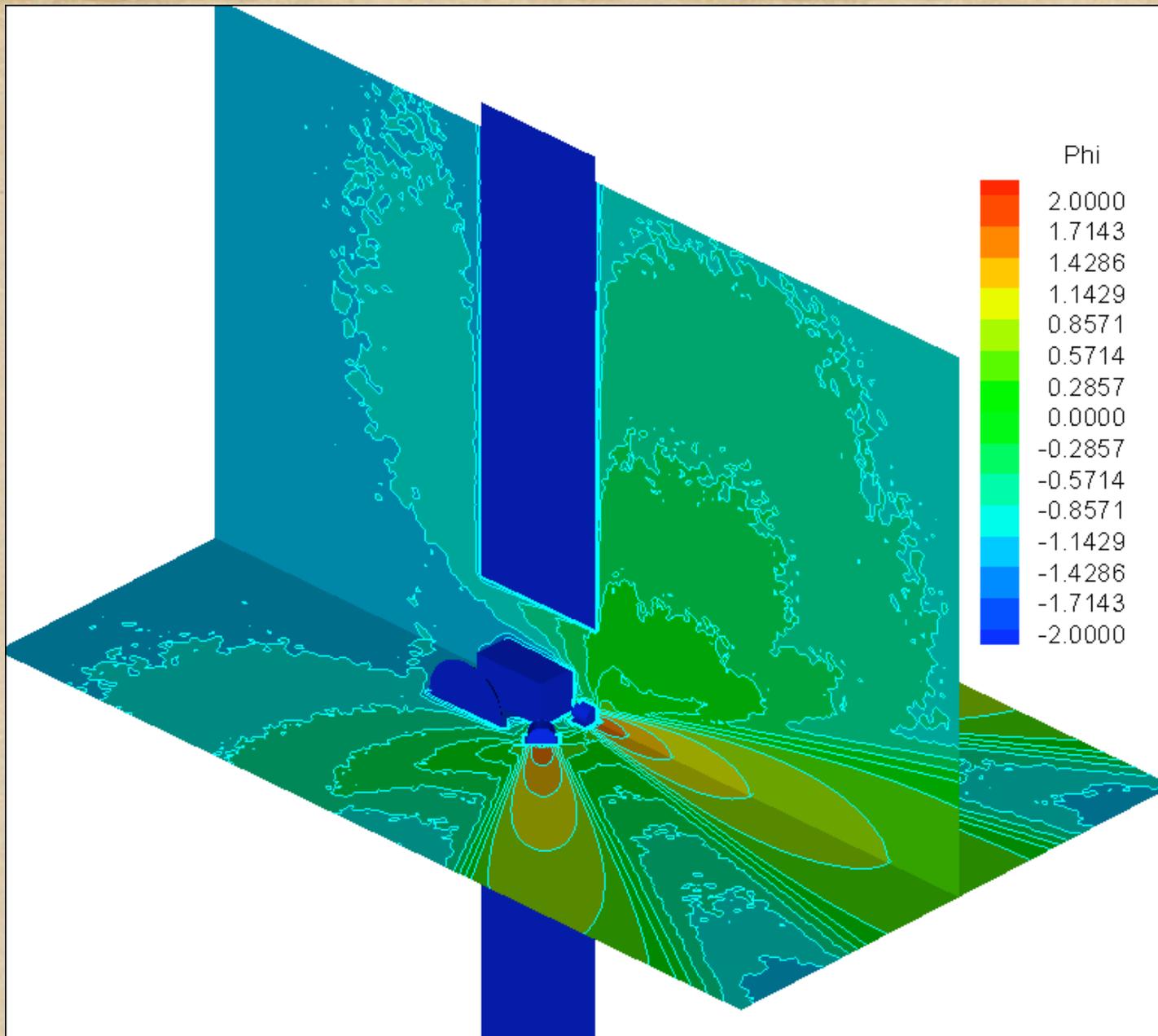
# Codes using UPIC Framework

**QuickPIC:** Quasi-static code for plasma based accelerators. QuickPic is Darwin in the transverse direction

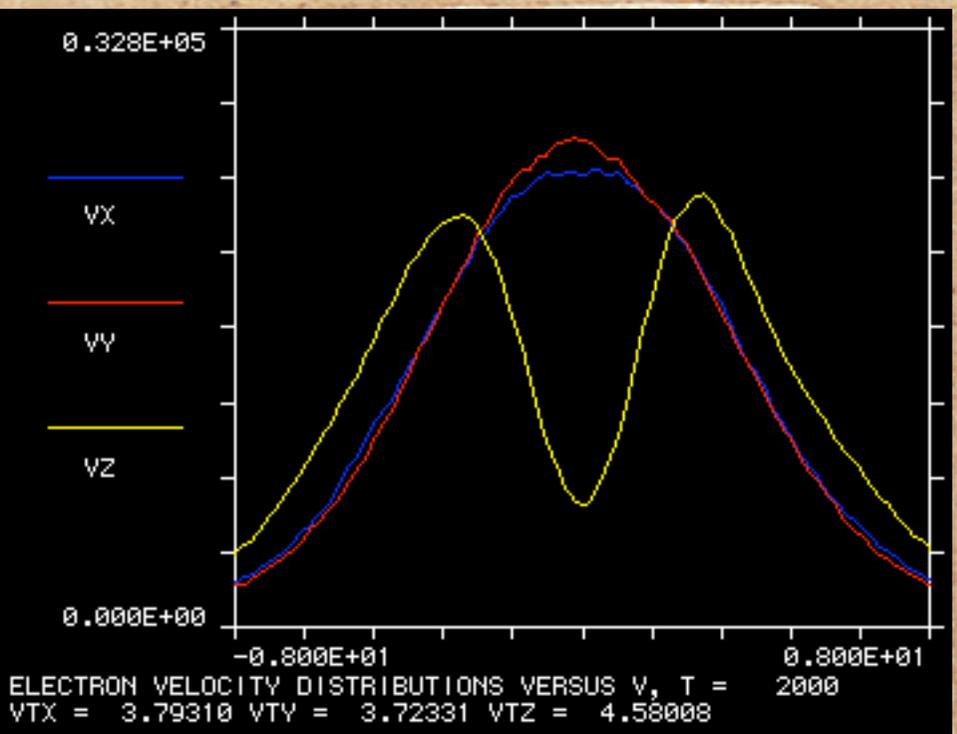
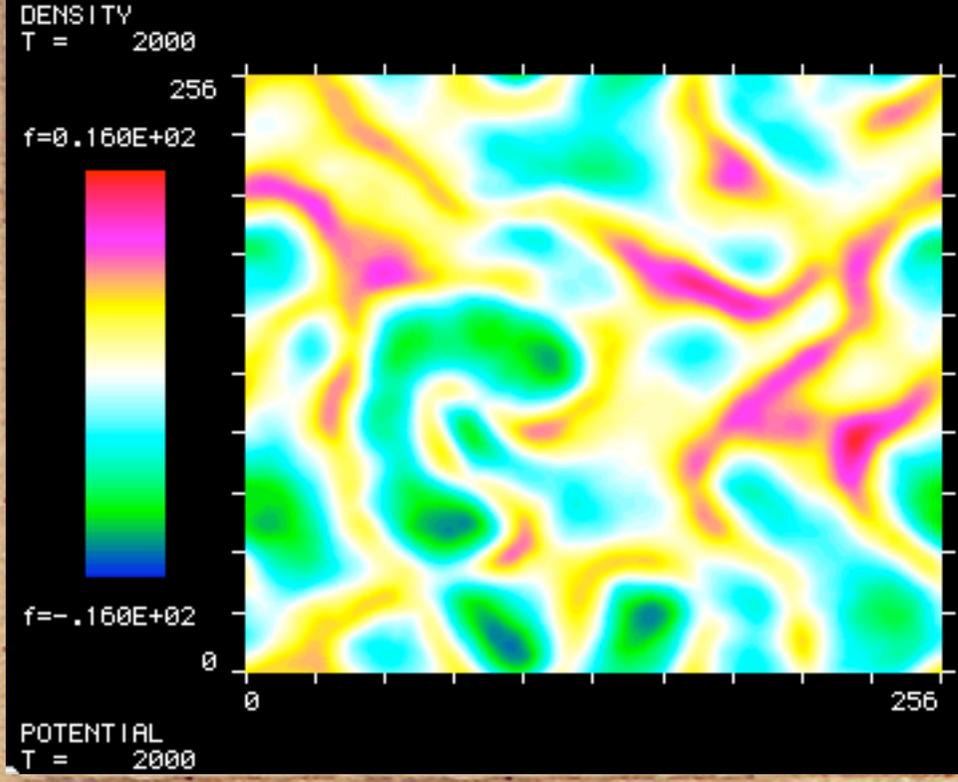
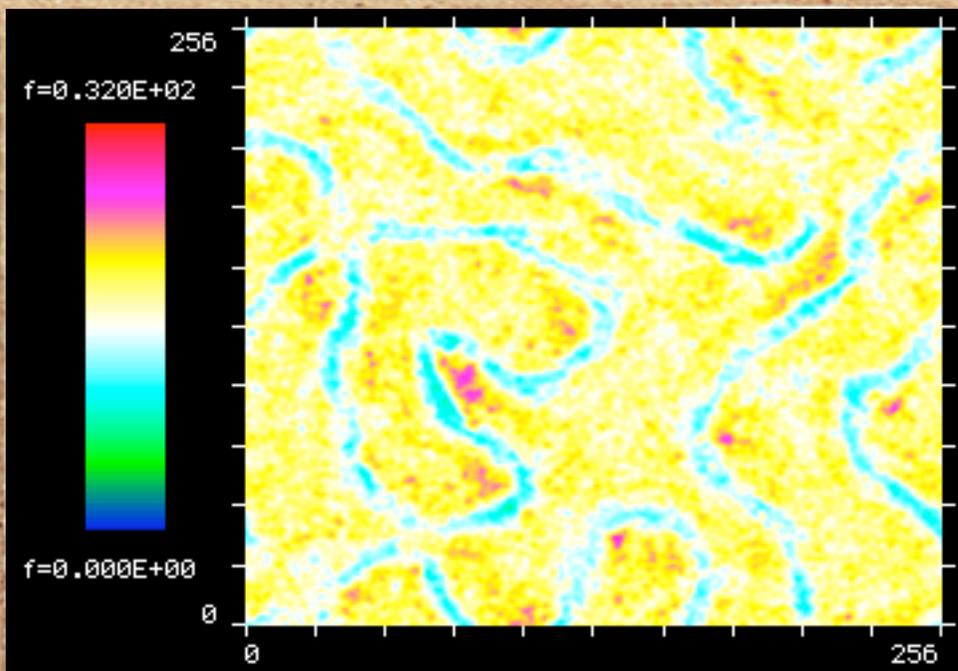
**QPIC:** Quantum PIC codes using Feynman path integral formulation

**Parallel DRACO:** 3D code for modeling ion propulsion. Major area of interest is contamination of spacecraft and sensors.

**BEPS:** an interactive 2D/3D PIC code for teaching plasma physics. Used by graduate student to model symmetric neutralized ion beams for fusion applications



DRACO: Ion Propulsion, J. Wang, et. al.



2D BEPS Code:

Relativistic  
Electromagnetic Two  
Stream Instability