FLASH, a Modern, Well Tested, Multiphysics Application Code that Scales from Laptops to the Largest Supercomputers

Anshu Dubey
June 22, 2009
The FLASH Code Contributors

- **Current Group:**
  - Klaus Weide, Chris Daley, Lynn Reid, Paul Rich and Anshu Dubey

- **Other Current Contributors:**
  - Dongwook Lee, Paul Ricker, Dean Townsley, Cal Jordan, John Zuhone, Kevin Olson, Marcos Vanella

- **Past Major Contributors:**
  - Katie Antypas, Alan Calder, Jonathan Dursi, Robert Fisher, Timur Linde, Tomek Plewa, Katherine Riley, Andrew Siegel, Dan Sheeler, Frank Timmes, Natalia Vladimirova, Greg Weirs, Mike Zingale
The FLASH code

1. Parallel, adaptive-mesh refinement (AMR) code
2. Block structured AMR; a block is the unit of computation
3. Designed for compressible reactive flows
4. Can solve a broad range of (astro)physical problems
5. Portable: runs on many massively-parallel systems
11. Scales and performs well
12. Fully modular and extensible: components can be combined to create many different applications
The ASC/Alliances Center for Astrophysical Thermonuclear Flashes
The University of Chicago

FLASH Users Community (2007 survey)

Flash Code Survey Responses

Breakdown of Responses

- Not using: 17%
- Sample code/concept testing/educational: 25%
- V&V: 9%
- Haven't used yet, plan to in future: 8%
- Primary research tool: 41%
FLASH Performance

Mean clock cycles to complete 10 evolution steps - I/O switched off
(Number of leaf blocks / processor kept approximately constant in each experiment)

The ASC/Alliances Center for Astrophysical Thermonuclear Flashes
The University of Chicago
FLASH Basics

- An application code, composed of units/modules. Particular modules are set up together to run different physics problems.
- Fortran, C, Python, …
  - More than 500,000* lines of code, 75% code, 25% comments
- Very portable, scales to tens of thousand processors

Capabilities

- Infrastructure
  - Configuration (setup)
  - Mesh Management
  - Parallel I/O
  - Monitoring
    - Performance and progress
  - Verification
    - FlashTest
      - Unit and regression testing

- Physics
  - Hydrodynamics, MHD, RHD
  - Equation of State
  - Nuclear Physics and other Source Terms
  - Gravity
  - Particles
  - Material Properties
  - Cosmology
Auditing Process

- SVN for Version Control
- Test Suite
- Online Coding Violation Tracking and Bugzilla
  - Unfinished tasks, bugs, bad code, developer queries
- Profiling Tools
  - Memory / speed diagnostic tools
  - External tools like JUMPSHOT / PAPI / TAU
- Documentation
  - Online documentation for Unit APIs -- ROBODOC
  - User’s guide in HTML and PDF
  - “Howto” available for developers, various platforms
  - Email users’ group
FLASH Units

- FLASH basic architecture unit
  - Component of the FLASH code providing a particular functionality
  - Different combinations of units are used for particular problem setups
  - Publishes a public interface for other units’ use
  - Can have more than one subunit
  - Can have multiple alternative implementations, including null implementation
  - Individual routines can be customized

- Inheritance through configuration tool and directory structure

- Interaction between units governed by the Driver

- Not all units are included in all applications
FLASH Setup Script: Implements Architecture

Python code links together needed physics and tools for a problem

- Parses Config files to
  - Determine a self consistent set of units to include
  - If a unit has multiple implementations, finds out which implementation to include
  - Get list of parameters from units
  - Determines solution data storage

- Configures Makefiles properly
  - For a particular platform
  - For included Units

- Implements inheritance with unix directory structure

- Provides a mechanism for customization
Runtime Environment

- Collection of all “Parameters” declared in all the Config files parsed by the setup.

- File “setup_params” generated by the setup contains all runtime parameters found, and their initial value.

- The initial values are picked from Config files. They can be overwritten by including them in “flash.par”
Infrastructure: Mesh Packages in Flash

**Paramesh**

- Block Structured
- Fixed sized blocks
- Specified at compile time
- Not more than one level jump at fine coarse boundaries

**Uniform Grid**

- One block per proc
- No AMR related overhead

The ASC/Alliances Center for Astrophysical Thermonuclear flashes
The University of Chicago
I/O Libraries

- FLASH works with 2 different I/O libraries
  - HDF5
  - Parallel-NetCDF
- Use MPI-IO mappings
- Both Portable libraries
- Scientific Data mostly stored in multidimensional arrays

- FLASH3 also supports a basic direct FORTRAN I/O -- use only as a last resort!